Yield Prediction with Explainable Machine Learning

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von
Florian Philipp Huber
aus
Mechernich

Bonn 2024

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

Gutachter / Betreuer: PD Dr. Volker Steinhage
Gutachterin: Prof. Dr. Elena Demidova

Tag der Promotion: 31.10.2024
Erscheinungsjahr: 2024

Abstract

Starting from a federal project to predict grapevine yields in Germany, we faced five challenges to enable machine learning for yield prediction. The first challenge is training on small data sets, as capturing data for yield prediction is very time consuming with most plants following an annual cycle. Providing a feature-based representation of remote sensing data by modeling underlying distributions allows gradient boosting methods to outperform deep learning approaches by 25% in our experiments for soybean yield prediction in the US, one of the biggest datasets for yield prediction that allows international comparability. The second challenge is the need for explanations to show that the model's decision making is in-line with experts knowledge of the field. For this challenge, we extend the idea of Shapley value feature attributions to predefined groups of features. The groupings are naturally given for yield prediction scenarios and allow for an improved representation of the explanations, as individual features are plentiful and often abstract. We give a novel algorithm to solve the problem of calculating the grouped Shapley values in polynomial time for random forests as they result from the gradient boosting pipeline from challenge one. Third, we work towards better feature selection for yield prediction tasks. The introduction of grouped Shapley values sparks the question of whether Shapley values could be used for feature selection. To address this question, we define four necessary conditions for defining a Shapley value suitable for feature selection. Additionally, we analyze the problem of model averaging where unimportant features are allowed to alter the final feature selection by introducing a novel exhaustive feature selection tool that has no problems with model averaging, and use it to further evaluate Shapley values for feature selection. Our experiments indicate that there is a small loss in accuracy due to model averaging, while the runtime of Shapley values as a heuristic measure for feature selection is superior for random forests. The fourth challenge is handling gaps in remote sensing data. As we need to use remote sensing data to provide consistent coverage for a small research area, clouds that occlude the satellite's view on the Earth can hide a meaningful amount of data. We approach this challenge by introducing a novel deep interpolation pipeline that uses a U-Net structure together with partial convolutions to gradually fill in remote sensing data in our research area, finally improving previously established statistical methods by 44% in terms of RMSE. Lastly, we worked towards a solution to make predictions for shifting domains, where we used regularized transfer learning to improve yield prediction by transferring knowledge between different domains by 16% in terms of RMSE, compared to not using transfer learning techniques.

Acknowledgments

Foremost, I would like to express my deepest gratitude to PD Dr. Volker Steinhage for offering me the opportunity to work in his research group and for providing invaluable guidance and advice throughout the entire process. His support has been instrumental in the completion of this thesis.

I am also grateful to Frank and Timm, with whom I shared an office, for their constant support, numerous discussions, and all the proofreading of my papers. Their collaboration and encouragement have been greatly appreciated.

Special thanks go to all the students who worked with me on the KI-iREPro project or trusted me with guidance during their bachelor thesis. Artem, Alina, Alvin, Benedikt, Dylan, Kai, and Stefan, your contributions and hard work have significantly enriched my research and this thesis.

I would also like to extend my thanks to all the members of the KI-iREPro project. In particular, Anna, Benedikt, Hannes, Katja, Michael, and Robin, your partnership and insightful contributions have been invaluable, especially when explaining viticulture processes patiently.

The project was supported by funds of the Federal Ministry of Food and Agriculture (BMEL) based on a decision of the Parliament of the Federal Republic of Germany via the Federal Office for Agriculture and Food (BLE) under the innovation support programme.

My sincere gratitude goes to the committee of my PhD thesis, including Prof. Dr. Elena Demidova, for providing guidance and advice, and to the other committee members for their time and effort in reviewing my work.

On a personal note, I am deeply thankful for the unwavering support of my friends and family. Foremost among them are my fiancée, Antonia, who supported me no matter the circumstances, and my best friend, Maurice, who always listened to new ideas and was ready to celebrate any accomplishment. Your encouragement and belief in me have been a source of great strength and inspiration throughout this journey.

Thank you all.
Florian

Contents

1 Introduction

Providing a secure source of food for the world's population is an increasingly difficult challenge. While the world population grows, the available farmland around the world is limited, driving us to optimize agricultural efficiency to its full extent. Doing so will be an integrated task of agriculture and political will, along with logistics of food distribution around the world (Prosekov & Ivanova 2018). To be able to make informed decisions, we need to be able to provide resilient data for the amount of food available, which, in the end, is a question of predicting the yearly yields for a variety of different crops around the world. The accurate prediction of yields then comes with multiple benefits, such as advanced logistic planning capabilities, the possibility of redistributing food throughout the world, and financial security for producers. Within this thesis, our aim is to provide tools for reliable and exact yield prediction using explainable machine learning algorithms.

Machine learning in yield prediction has developed an increasingly high accuracy over time (Van Klompenburg et al. 2020) and has subsequently replaced statistical modeling approaches to predict crop development and yields. The advantages of machine learning models over statistical approaches are multiple. For one, statistical methods often rely heavily on experts' understanding of biodiversity, plant-specific properties, and local ecosystems (Lobell & Burke 2010). Changes in the environment will result in the change of variables during the modeling process and ultimately in miscalculated predictions. Especially in times of climate change, it cannot be guaranteed that the environmental conditions remain stable. Machine learning methods used for yield prediction can circumvent the well-known problem to an extent, by being able to react to new environmental conditions and the resulting new patterns determining the yearly yields, by simply being retrained on newly acquired data, representing the novel conditions. However, with increasing accuracy, the complexity of the models and input data also expands. Popular data sources include remote sensing data (Huber et al. 2022, You et al. 2017), fertilization schedules (Meng et al. 2021), climate information (Cao et al. 2020), and soil evaluation (Sirsat et al. 2019), just to name a few. Successful models can utilize up to 1000 features to predict yield (Huber et al. 2022), making models and data increasingly difficult to understand. This is why we need to focus on explainable machine learning to not only provide accurate predictions but also provide explanations for the models decisions. Explaining the models decisions can raise trustworthiness by showing that the process of making predictions is in line with the knowledge of experts of the field, or by explaining the complex relations determining the yearly yields in general. Throughout this thesis, we will focus on the game-theoretic concept of Shapley values to obtain explanations for our models. Shapley values can provide a unified framework for explaining machine learning models with desirable properties. For tree models specifically, we can find a feature attribution score that is able to allocate credit among all features equally, regardless of the position in the tree. Moreover, Shapley values for ex-

plaining tree-based models are known to have strong performance on several benchmarks and consistency with human intuition (Lundberg et al. 2020).

1.1 The Challenges of Explainable Yield Prediction

Throughout the work in this thesis, we identified five major challenges to achieve reliable and explainable machine learning solutions for different yield prediction scenarios. In the following, we will explain the five challenges from different points of view. Each challenge in general can be seen as a universal challenge in machine learning, where regardless of the specific scope of the task at hand, we find similar problems in the literature. We then point out the specifications for the regarding challenge that arise from our scope of yield prediction. Finally, we will give an example of how the challenge impacted our work during the ongoing federal project "Künstliche Intelligenz für innovative Ertragsprognose bei Reben" (KI-iREPro). The title of the project can be translated to "Artificial Intelligence for innovative Grapevine Yield Prediction". Founded in April 2021, the project includes partner expertise in commercial wine growing, experimental breeding of new varieties, and computer science. The goal of the project is to enable accurate predictions of grapevine yields by incorporating extended data acquisition and finally machine learning-based yield prediction models into viticultural processes. To explain the challenges in this introduction, we give selected details of the project as necessary, with more information on the project described in Section 3.2. Following the challenge explanation, we will emphasize the measures taken to achieve an improved explainable yield prediction.

a) Training on small data sets

Suffering from a lack of training data to represent the underlying distribution of the scenario at hand is known to be a constant problem for machine learning tasks. Numerous solutions to this challenge exist, from generating synthetic data, over using unsupervised methods that eliminate the need for labels, to choosing machine learning models that require fewer data points to learn underlying relationships. Although many problems in machine learning are tackled by deep learning approaches, and neural nets are known to be an universal function approximator if enough data are available, they might not be the best choice for a machine learning problem with limited training data. For yield prediction, in general, the scarcity of data arises from being able to obtain new data points only once a year, during harvest. Even the largest data sets available, for example for soybean yield prediction in the US, where the US Department of Agriculture is a pioneer for accurate and reliable data acquisition in agriculture, can offer little more than 15,000 data points to work with. For deep learning applications, we often have access to more training data, such as the popular ImageNet data set that consists of 14 million annotated images (Deng et al. 2009). Furthermore, multiple data points for yield prediction taken from the same year always show some form of correlation, since growing conditions in a year only allow for so much variation; inducing bias into the model if too few years are considered during model training. Within the KI-iREPro project, we had access to little more than ten years of historical records that can be used as ground-truth yield data for creating a prediction model. Furthermore, data acquisition in the past was

not done with machine learning in mind. This, in conjunction with common agricultural practice in viticulture, results in artifacts, further shrinking the amount of data available. In this work, we present a pipeline to parameterize the underlying distributions of freely available remote sensing images to obtain feature-based representations that allow the use of eXtreme Gradient Boosting (XGBoost) for yield prediction (Huber et al. 2022), making reliable predictions with access to limited training data sets. We succeed in providing models with 25% higher accuracy in terms of RMSE, when compared to deep learning approaches, even for the largest and most widely used yield prediction data set within the field of research, soybean yield prediction in the US.

b) Need for explanations

Explainability is one of the most anticipated research directions in machine learning right now. According to Selbst & Barocas (2018), there are three general reasons behind the call for an explainable AI. (1) the fundamental question of autonomy, dignity, and personhood, (2) educating about how to achieve different prediction results, and (3) explaining the model allows a debate on whether the model's rules are justifiable. Translating the motivation to yield prediction, the first point of motivation remains important even when we make no predictions directly related to individual human beings, as the yearly harvest cycle significantly touches on the life of the producers. Similarly, it is desirable to learn how to maximize annual yields and increase trust in predictions by providing a way to justify the model by experts. In KI-iREPro, we faced the challenge of explainability, especially with regard to the last talking point, with many viticulture experts in our project. It was desired to explain the yield prediction models in a way that both computer scientists and domain experts could find a common language. Especially challenging in explaining yield prediction models is the variety of input features that can be used to obtain yield predictions, bringing common explanation methods to the edge, as explanations are often confusing when too many input variables are explained at once. A solution to this problem can be given by grouping the explanation of features together, which are generally explained as one unity. This includes multiple features acquired by the same sensor or at the same point in time. We extend the well-known framework of Shapley values, giving a definition of grouped Shapley values that can conserve the additivity of explanations for groups of features, one of the key qualities of Shapley values. Building on existing work for fast calculation of Shapley value feature attributions, we give an algorithm for polynomial calculation of grouped Shapley values for random forests. Exploiting the tree structure, we calculate the final Shapley value by adding the contributions of each path in the tree, instead of following the direct definition of grouped Shapley values, where an exponential amount of calculations is needed. This is possible since each path of a tree is included in multiple steps of naive Shapley value calculation. We can finally use our approach to explain the machine learning models that are created to solve the first challenge described above.

c) Selecting important features

The first step in designing a machine learning model is to define the data that should be captured for the predictions. While for some problems finding the right input features

comes naturally, there are others where the number of possible input features is nearly endless and we need to decide on a meaningful representation of the data. The task of feature selection is historically divided into three different families (Huan & Hiroshi 1998). First, the filter approaches where the feature subset is selected based on some quality measurement that is independent of any machine learning algorithm. Second, wrapper methods, where the feature selection is wrapped around a machine learning algorithm to generate a feature subset based on the algorithms' performance. Then we have the third family, embedded methods, where we directly compute the importance of the features based on their contribution within the machine algorithm. A famous example for embedded feature selection for random forests is a greedy feature selection based on the internal Gini importances of random forests (Menze et al. 2009). With regard to yield prediction, annual yields are determined by a multitude of different influencing factors. Climatic conditions, soil properties, and agricultural practices are just some examples. The underlying relations that determine the yields from these factors are very complex and difficult to model. Data that can be used for yield prediction are plentiful, but in-field measurements by experts or specialized sensors are an excellent source for precise predictions, but are often expensive to obtain. Within KI-iREPro, we noticed that data acquisition for yield prediction often follows a pattern, where the same sensor usage or expert assessments take place multiple times throughout the year. Reproducing a models success with fewer input features could lead to fewer sensors or expert hours needed to obtain future predictions, ultimately increasing the chance of real-world adoption of our yield prediction models. For our feature selection approach, we build on the concept of grouped Shapley values. With Shapley value feature attributions being fast to calculate for random forests and offering accessible explanations, it is an open question whether we can leverage the attributions towards selecting important features. We contribute to the discussion by showing that a properly defined Shapley value feature attribution can solve most of the existing concerns regarding Shapley values for feature selection, while separating model averaging as an unsolvable remaining problem (Huber & Steinhage 2024d). In feature selection for prediction models, it is uncertain which features should be considered for the final feature selection. Model averaging is the process of quantifying the influence of all features with respect to predictions (Madigan & Raftery 1994, Raftery 1995). Finally, model averaging allows unimportant features to alter the feature selection, giving nonoptimal solutions. To analyze the impact of model averaging on real-world feature selection problems, we introduce a feature selection method based on estimating the expected model output for random forests for a restricted feature space. Analyzing all feature subsets naturally results in an exponential runtime of the approach, but allows for a comparison with Shapley value feature selection for reasonable sized problems. Our experiments only demonstrate a slight increase in performance when model averaging is mitigated, justifying Shapley values for feature selection given their superior runtime for random forests. Finally, using Shapley values for feature selection allows us to decrease the number of input features by 72% while only sacrificing 1% in accuracy, compared to our initial experiments for the prediction of soybean yield in the US.

d) Gaps in remote sensing data

Remote sensing data is a term used to refer to data captured by satellites orbiting the Earth. For example, NASA satellites Terra and Aqua equipped with the Moderate-Resolution Imaging Spectroradiometer MODIS (Vermote, E. 2021) capture daily information about visible and infrared reflectance of the Earth's surface that can be used to infer the phenological condition of plants (Gao 1996). They provide an important data source for a variety of machine learning applications, such as monitoring changing climate (Schneider & Hook 2010) or analyzing the environmental impacts of land cover (Gohain et al. 2021). Furthermore, they play an important role in yield prediction, where they offer a reliable and accessible data source to match historical records of yield quantities with climatic and phenological information about the plant state at the time, even though the data suffer from missing information induced by clouds blocking the satellite's view on the research area. We witnessed this problem during data preparation for grapevine prediction during KI-iREPro. Current solutions include either using remote sensing data at a low spatio-temporal resolution to avoid gaps or the interpolation of the gaps based on statistical methods. However, plot-level predictions require the use of high-resolution satellite data, and statistical models are often unable to learn the underlying climatic pattern of the interest regions. Within this thesis, we introduce a deep interpolation practice to fill gaps in remote sensing Land Surface Temperature (LST) data with the help of local ground-site weather stations and a U-Net architecture incorporating partial convolutions, resulting in a full coverage of our study area used within KI-iREPro. This is done with a novel deep learning pipeline that can learn on partial ground-truth data exclusively. This is possible by (1) performing partial convolutions, gradually filling unknown pixels with information based on their neighbors, and (2) defining a loss function and training pipeline that evaluate the reconstruction only on valid pixels (Huber et al. 2024b). By doing so, we allow model training, even when the amount of non-occluded ground-truth data is severely limited. The improvement in reconstructing the daily Land Surface Temperature (LST) in our study area is measured by a decrease of 44% in terms of RMSE compared to state-of-the-art statistical modeling.

e) Making predictions for shifting domains

All throughout machine learning, researchers are faced with changing data properties, before or during the application of a machine learning model (Zhou et al. 2022). Most machine learning algorithms are highly based on the assumption that the data used for training and the data found during the application are drawn from the same distribution. But this is not always the case, especially when domains shift in some kind of form. Yield prediction scenarios themselves are quite versatile, including different regions around the world and different crops and crop varieties. While some scenarios are extensively explored and data acquisition is already occurring over an extended time period, others are not. Naturally, the question arises if our models can handle the change in domain from one yield prediction scenario to another, for example by changing the growing region or the crop variety at hand. Traditionally, the easiest solution to this problem is to retrain whole machine learning models with a more fitting training data distribution.

However, this way of solving the problem has the potential to hold the approaches back, as they cannot integrate previous learning from a different domain into the pipeline. In this thesis, we explore the possibilities of transfer learning by examining soybean yield in the US. For this, we use deep learning instead of XGBoost, as decision tree approaches to transfer learning (Segev et al. 2016, Jiang et al. 2019) are not as powerful, especially when the respective domains are inhomogeneous. We use a Convolutional Neural Net (CNN) to learn features from remote sensing satellite data for soybean yield prediction in the US and transfer the knowledge to soybean yield prediction in Argentina (Huber et al. 2024a). For this, we explore the capabilities of different state-of-the-art regularization techniques and attach a deep Gaussian process to improve the results, revealing an improvement of 16% in terms of RMSE for soybean yield prediction in Argentina when using knowledge transferred from soybean yield prediction in the US, compared to not using transfer learning.

1.2 Contributions

This work presents five distinct contributions that correspond to the five challenges for yield prediction identified in the KI-iREPro project, as previously mentioned. While the solutions are formulated with yield prediction in mind, the ideas for explainability and feature selection extend beyond the horizon of yield prediction and can ultimately be useful in a variety of machine learning tasks. Each contribution can be linked to a specific peer-reviewed publication on the topic. The summarized contributions are as follows:

a) Extreme Gradient Boosting for Yield Estimation Compared with Deep Learning Approaches (Huber et al. 2022)

- **Conceptual:** Analyzing gradient boosting methods for yield prediction on remote sensing data compared to deep learning approaches.
- **Methodical:** Extracting features from remote sensing data in tabular form by modeling the underlying skewed normal distributions based on histogramization of remote sensing data.
- **Results:** Improving yield prediction for the biggest available data set — the soybean yield prediction in the US — by 25% in terms of RMSE over the state-of-the-art deep learning approaches.

b) Grouping Shapley Value Feature Importances of Random Forests for explainable Yield Prediction (Huber et al. 2023)

- **Conceptual:** Exploit given natural groupings of features from yield prediction scenarios to improve the usability of Shapley values to explain yield prediction models.
- **Methodical: 1.** Giving a polynomial algorithm to calculate the grouped Shapley values for random forests. **2.** Providing proof that the sum of individual Shapley values is not the same as using grouped Shapley values when calculated on random forests by giving a counterexample.

- **Results:** We provide Grouped Shapley Values (GSV) as a specialized tool for explaining yield prediction models directly attaching the method to the success of individual Shapley values and are therefore able to improve the explainability for yield prediction scenarios.

c) Conditional Feature Selection: Evaluating Model Averaging when Selecting Features with Shapley Values (Huber & Steinhage 2024d)

- **Conceptual:** Asking the research question if Shapley values should be used as a feature selection tool and evaluating theoretical shortcomings against performance in application.
- **Methodical: 1.** Defining four necessary conditions for defining a Shapley value suitable for feature selection. **2.** Introducing a novel exhaustive feature selection method (CFS) to evaluate the lost performance of using Shapley values for feature selection due to model averaging. The core idea is to evaluate the expected model output with limited access to feature subsets.
- **Results:** The baseline approach of greedily selecting features according to the internal Gini feature importance of random forests is consistently improved on by both, a feature selection according to Shapley values by circa 5% in terms of RMSE and according to CFS by circa 7.5% in terms of RMSE when averaged over all possible subset sizes.

d) Deep Interpolation of Remote Sensing Temperature Data with local Weather Stations and Partial Convolutions (Huber et al. 2024b)

- **Conceptual:** Carrying over the success of deep learning for image inpainting towards interpolating gaps in remote sensing land surface temperature (LST) data.
- **Methodical: 1.** Design of a U-Net architecture specific to our region of interest. **2.** Introduction of a training pipeline allowing the exclusive use of partial ground-truth images for model training where two partial images are combined to create realistic cloud occlusions, including the introduction of a partial loss function.
- **Results:** Improvement over previous state-of-the-art statistical methods by 44% in terms of RMSE while filling 100% of all pixels.

e) Leveraging Remote Sensing Data for Yield Prediction with Deep Transfer Learning (Huber et al. 2024a)

- **Conceptual:** Answering the question whether regularized transfer learning is possible for yield prediction scenarios with widely differing domains.
- **Methodical: 1.** Data handling to achieve spatio-temporal alignment of the two data domains. **2.** Design of a deep learning set-up which allows transfer learning using established building blocks including frozen weights and regularization (L2-SP and BSS) driven fine-tuning followed by the application of a Gaussian process.
- **Results:** Improvement for soybean yield prediction of 16% in terms of RMSE for Argentina with transfer learning compared to CNN training without transfer learning.

1.3 Thesis Structure

The following thesis is generally set up in the well-known manner of a research paper, consisting of Related Work, Materials, Methods, and Experiments. Finally, we add a chapter showing the practical usage of the methods as they are used in the real-world by our partners within the KI-iREPro project. The chapters Related Work, Methods, and Experiments consist of five sections, each dedicated to one of the five contributions, as shown above. Most of the content in the chapters has been published in peer-reviewed journal and conference contributions. For international comparability, we will first test our pipeline for yield predictions with the well-renown soybean yield records captured from the US Department of Agriculture (USDA), one of the largest data sets available for yield prediction. The prediction model will be explained using grouped Shapley values and will be optimized by feature selection. In the following, we apply the model to the commercial and breeding aspects of the KI-iREPro project, first fixing the gaps in remote sensing LST data in our research area. Lastly, we analyze the transferability of knowledge in the context of yield prediction, again with regard to soybean yield data, since for this task the amount of available data allows training expressive deep learning models that will serve as the basis for transfer learning efforts.

2 Related Work

In the introduction, we established five challenges for yield prediction. Each of the challenges is suspect of previous research, which will be displayed below and marks the starting point of this thesis contributions. As we have addressed each of our contributions to the five challenges in a peer-reviewed publication, we will base the related work for each challenge on one of our publications, extending the related work with new additions if necessary. We start by explaining how the task of yield predictions and subsequently the problem of limited data are handled in the literature. We then describe efforts to obtain explainable models in yield prediction scenarios and introduce the history of Shapley values in machine learning. We present related work discussing feature selection, first in general, and then with the help of Shapley values. After this, we explore how others solved the problem of gaps in remote sensing data and, lastly, we will explain the related work regarding the transfer of knowledge between different models in yield prediction, as it is necessary to handle shifting domains.

2.1 Yield Prediction

The topic of yield prediction is extensively covered in the surveys of Van Klompenburg et al. (2020) and Nathgosavi & Patil (2021). For the sake of this thesis, we focus on the discussion by dividing the research field into three types of approaches to yield prediction. First, solutions based on statistical models. Second, machine learning approaches without deep learning and (3) solutions based on deep learning. This section is based on previously published work (Huber et al. 2022).

Statistical Models for Yield Prediction

As yield prediction is a research topic throughout history, we find a research branch established before the widespread use of machine learning. General statistical models with adjustable parameters are fitted to the specific yield prediction problem at hand, often under the consideration of domain experts, helping with establishing the parameters. Two representative approaches to yield prediction based on statistical models and vegetation indices are reported by Meng et al. (2019) and Zhao et al. (2020). Meng et al. (2019) propose a new daily vegetation index based on MODIS data and use it to predict cotton yield in California. An exponential function with two learned parameters is used to predict the yields. They achieve good results for yield prediction on a field scale by combining images with low and high spatial resolutions. Zhao et al. (2020) predict wheat yields at a field scale in Wales with traditional crop modeling based on high-resolution images from the Sentinel-2 satellite. Their yield prediction model was developed using statistical analysis of variance (ANOVA) and a multivariate analysis that incorporates various derived metrics and indices. Statistical models are also used

in grapevine prediction. De La Fuente et al. (2015) report good results for grapevine yield prediction when counting field measurements such as the number of clusters and the weight of the cluster in different time frames throughout the growth cycle of plants and predict the expected yield by comparing the reported feature values with historical values of the same feature in previous years.

While these approaches are data- and computationally efficient, the handcrafted modeling of multiple complex interactions with statistical models is very rigid and vulnerable to not scaling well with changes in the environment. Therefore, the XGBoost-based approach presented in this thesis aims to benefit from the intrinsic advantages of machine learning approaches, that is, being adaptable to changes in the environment by retraining using new training data.

Machine Learning for Yield Prediction

Machine learning methods often improve the performance of statistical methods and, most importantly, can be implemented without expert knowledge, using a sufficient amount of training data instead. The works of Rodríguez et al. (2017) and Bóbeda et al. (2018) focus on predicting the yields in citrus orchards. Both approaches use a selection of phenological information that is captured directly from plants multiple times a year. For both, the implementation of M5-Prime (Wang & Witten 1996) of random forests shows superior results on different kinds of citrus fruits. Sirsat et al. (2019) describe an approach to grapevine yield prediction using phenological information, soil properties, and climatic conditions. The random forest ensemble technique proposed by Breiman (2001) shows the best results to predict grapevine yield. More recent publications show the benefits of incorporating remote sensing data for yield prediction. Martínez-Ferrer et al. (2020) include remote sensing data in terms of vegetation indices together with phenological information and use a Gaussian process to predict the yield of corn, wheat and soybeans in the US. Meng et al. (2021) employ vegetation indices for the prediction of maize yield in California on a field scale. Complementing data sources include fertilizer information, climate data, and soil properties. They conclude that non-linear models such as random forests are best performing for this kind of prediction task. The first study to use XGBoost for yield prediction is that of Charoen-Ung & Mittrapiyanuruk (2018). Based on the characteristics of the plot, the phenological information, and the rainfall volume, they predict the yield of sugar plots in Thailand. This approach forgoes an exact yield regression and instead solves the binary classification problem, whether the yield is above or below the median of the training data. Cao et al. (2020) deploy LightGBM as a form of gradient boosting for random forests to predict winter wheat yields in China. Again, they used vegetation indices, derived from remote sensing data, together with climatic and socioeconomic factors as input data. In recent advances, Desloires et al. (2023) achieved similar results with deep learning and XGBoost for the prediction of corn yield on the field scale in the US. Instead of using the regular way of aggregating the data according to a fixed time frame, they calculated the sampling pattern according to the actual growth periods of the plants to improve the model accuracy. Celik et al. (2023) achieved state-of-the-art results for cotton prediction using XGBoost and LightGBM.

The novelty introduced within our work for explainable yield prediction is the com-

bination of machine learning and our way of sampling remote sensing data to achieve a tabular representation, by exploiting a representation as skewed normal distributions of the features involved in the prediction.

Deep Learning for Yield Prediction

The latest developments in yield prediction include the use of deep learning architectures. The extensive use of deep learning for the prediction of yields gained momentum with the publication of You et al. (2017). They investigated the deployment of basic convolutional neural networks (CNNs) and long-short-term memory networks (LSTMs) for yield prediction on a USDA soybean data set similar to the soybean yield prediction data set examined in this thesis. Furthermore, they developed the idea of using binned histogramizations of remote sensing data to overcome the limitations that are implied by nonregular-sized images for deep learning. The same concept is picked up by Sun et al. (2019), who evaluate a CNN-LSTM hybrid model for the same yield prediction task and report improved accuracy. Other notable publications, including CNNs, are presented by Wang et al. (2020) and Wolanin et al. (2020). Forgoing complex structures such as CNNs and RNNs, the work of Khaki & Wang (2019) shows that an accurate prediction of maize yield based on environmental and genotype information is possible using only simple feedforward networks. They predict future weather conditions to obtain a more accurate prediction of yield in the early stages of the year. Alibabaei et al. (2021) demonstrate the benefits of incorporating the exact irrigation schedule of the crops examined. Given very detailed input information, they compare predictions made by several algorithms, including LSTMs, CNNs, and feedforward networks. The experiments show a superior performance of a bidirectional LSTM structure in predicting the yields of potatoes and tomatoes. The data provided by the USDA have become very popular in recent years because it is one of the largest sources of yield data. Like most regression tasks, yield prediction becomes more difficult when there are fewer training data available. Wang et al. (2018) address this problem by applying the concept of transfer learning. They first train an LSTM network for soybean yield prediction in Argentina and then transfer the knowledge to regions in Brazil that have fewer available data. Another idea to solve this problem is introduced by Khaki et al. (2021), who combine the predictions of different crops to overcome data scarcity. They present a multi-target regression CNN structure, together with a newly developed combined loss function. Experiments are again carried out on USDA data by simultaneously predicting soybean and corn yields based on remote sensing data, showing promising results.

There are two state-of-the-art approaches to soybean yield prediction based on deep learning that are explicitly demonstrated and evaluated on USDA and remote sensing data, namely the CNN-based approach of You et al. (2017) and the CNN-LSTM hybrid model of Sun et al. (2019). Later in this thesis, our approach to an XGBoost-based yield prediction system will be compared with these two approaches (Section 5.1).

2.2 Explainability and Shapley Values

Again, we will review the related work divided into categories. First, we will give an overview about explainability in machine learning as a whole. Then, based on the related

work section of one of our previous publications (Huber et al. 2023), we will examine the state of explainability in yield prediction. Lastly, we focus on a specific idea to obtain explainable machine learning in the form of Shapley values, as they will be an integral part of our explainability pipeline, we will describe the history of using Shapley values as feature importance measure together with previous efforts to expand the paradigms of Shapley values onto groups of features, both in a general game-theoretic and a machine learning context.

Explainability in Machine Learning

With the widespread adoption of machine learning and a rise in precision throughout a multitude of tasks, also the complexity of machine learning models increased. This naturally results in an increased interest in the scientific field of explainable artificial intelligence (XAI), with the aim of explaining the decisions of machine learning models in human understandable terms to raise trust in the predictions. A comprehensive survey of the topic is given by Linardatos et al. (2020). The approaches to explain models can generally be divided into two groups. First, methods providing local explanations, where the predictions for individual data points are explained, and second, methods for global explanations, where the model is explained without consideration of a specific data point.

The local explanation methods can then be divided into model-specific and model-agnostic approaches. While model-specific approaches are designed to explain a specific kind of machine learning model, model agnostic approaches can be applied universally to any kind of machine learning model. In recent years, the most famous model-specific approaches have been developed for deep learning approaches, where the black-box nature of models with ever-increasing complexity are in desperate need of explanations. Deep learning specific approaches often perform some kind of backtracking of neuron activation to see which input neurons have the highest impact on the models' prediction. The idea was first proposed by Simonyan et al. (2013), where a gradient explanation technique was introduced. Gradients were used to calculate how small changes in the input will change the model output, and input pixels with high gradients are seen as important to the model's decisions. This main idea was improved many times, for example by Sundararajan et al. (2017). They achieved the desired properties of explanations by changing the method to calculate the gradients. Another approach for explaining deep learning approaches are Class Activation Maps (CAMs) as first introduced by Zhou et al. (2016) to achieve explainability for CNNs. CAMs indicate discriminative regions of an image that are used to determine the classification result of the network. Selvaraju et al. (2017) extend on this idea with the so-called Grad-CAM approach. Grad-CAM stands for gradient-weighted class activations mapping and uses the gradients of the target flowing into the final layer of a CNN to produce localization maps that highlight the important regions in the input image, finally allowing explanations for any kind of CNN. Model-specific explanations also exist for other models, so for example random forests. Mollas et al. (2019) introduce local explanations, where they determine the feature ranges, where a change in input will not change the output of the model. An extension of the work for multi label classification is given by Mylonas et al. (2022).

In comparison, there are other local explanation approaches that are model agnostic with an idea that can be applied to any machine learning model, allowing for unified

explanations. Famously, the LIME method for local model-agnostic explanations gained a lot of recognition (Ribeiro et al. 2016). LIME explanations are achieved by sampling data around the neighborhood of the input instance and using the predictions of the original model to train a smaller natively explainable model, like a decision tree. Giving explanations to the small model then automatically explains the source model. However, there are possible problems with the approach, for example, when poor parametrization leads to missing explanations (Garreau & Luxburg 2020). Evolving on the idea of LIME, Zafar & Khan (2019) propose a deterministic version of the approach, where the explanations are the same for every calculation by replacing the random sampling with a deterministic selection of neighboring data points via clustering. In another effort to create model agnostic explanations, the authors of LIME introduced anchors to explain models (Ribeiro et al. 2018). An anchor is a locally computed sufficient set of features and conditions such that the model output will remain the same. A game theoretic approach to achieve local model-agnostic explanations is given by the idea of Shapley values. We will discuss the related work regarding this topic in detail later on. A direct explanation of the model output is not the only way to achieve explainable machine learning. Another idea is contrastive explanations, where any model is explained by deciding which features are important such that a specific model answer is not given. This should help explainability in different domains, such as healthcare care and criminology (Jacovi et al. 2021). Wachter et al. (2017) describe an approach to find the smallest possible change that can be applied to the input features to achieve a changed output from the model. This creates counterfactual examples that can serve as a lightweight first interpretation of the model.

Global explainability is harder to obtain than local explanations, as the complex decision process needs to be described in general and not only with a specific example in mind. Friedman (2001) proposed the idea of Partial Dependence Plots (PDPs). They show how a certain set of features impacts the model's prediction by marginalizing the rest of the features. As a result, PDPs usually do not account for all features of the model and provide simplistic explanations. As an extension of PDPs, Goldstein et al. (2015) proposed ICE plots to address some weaknesses of PDPs by again restricting to local explanations. For random forests specifically, we also find a variety of global explanation approaches, since the underlying structure of decision trees is often times humanly understandable. Most famously, we refer to the Gini feature importance score, where the features are evaluated according to the amount of uncertainty they remove with respect to the splits inside the random forest (Nembrini et al. 2018).

Explainability in Yield Prediction

As described above, yield prediction as a research problem is being addressed with a multitude of approaches, both for modeling and explaining the results. We already described many deep learning approaches to the problem of yield prediction; for example, the work of You et al. (2017), Wang et al. (2020), and Khaki et al. (2021). Although gaining explanations of the models is difficult due to the black-box nature of deep learning models, some efforts were made to explain the results. You et al. (2017) correlate the importance of a feature with the decrease in the accuracy of the model when information on the feature is missing. They render the information of a feature useless by

randomly permuting the values throughout the data set. With this approach, they can assess the importance of whole feature groups by permuting their values simultaneously. Experiments show that the red and near-infrared bands of satellite images are important for their yield prediction model. Another way to gain explanations for the output of deep learning frameworks is deployed by Khaki et al. (2021). They backpropagate the output of active neurons in the last layer and are able to find active neurons in the first layer to correspond to features of the input space. An explainable alternative to deep learning for yield prediction can be found using tree structures like random forests, that are prevalent in yield prediction. When not using deep learning, we find other solutions to explain models in the literature. Díaz et al. (2017) use the M5-Prime algorithm (Wang & Witten 1996) to create regression trees for the prediction of citrus orchards in Argentina. Since singular trees are considered instead of forests, they conduct a visual analysis of the feature importances based on the resulting tree structure. Similarly, Bóbeda et al. (2018) use the M5-prime algorithm to predict citrus orchards in Argentina. They use another popular method to understand the yield predictions and explain their model output. By creating multiple subsets of features and evaluating their model in the absence of each of the subsets, they find that it is not necessary to count the fruits in the trees multiple times a year, and the results are only slightly worse when, instead, calculating the volume of the trees' crowns once. Since we can understand how tree models are functioning, we can rely on the inner relations of the trees to find feature importances that can give explanations to the model's output. The Mean Decrease in Impurity (MDI) (Louppe et al. 2013) can be used to give a measure of the number of splits made by each feature, weighted with the impact of the individual split, that is, the proportion of training samples divided. This internal measure of importance is used by Sirsat et al. (2019) to select expressive features when predicting grapevine yields based on phenological information, soil properties, and climatic conditions. Meng et al. (2021) use this method to show the high importance of vegetation indices when predicting maize yield in California on a field scale. For the prediction of cotton yield in the US, Celik et al. (2023) want to improve the explainability of XGBoost and LightGBM by using explainable boosting machines (EBM) introduced by Nori et al. (2019). By restricting the models to univariate terms and not considering interactions between the features during modeling, EBMs allow for better explainability traded off against the models' capacity. The work of Celik et al. (2023) uncovers precipitation as the most important variable for the prediction of cotton yields by using EBM, although it cannot beat the predictive performance of comparative gradient boosting models.

Our work within this thesis will be based on the explainability of random forests, where each individual tree can be easily interpreted by humans. Within this thesis, we will consider multiple features as inseparable a priori determined groups of features, as is always possible due to the time-series aspect of yield prediction. We combine this idea with the context of Shapley values as a universal tool for explaining machine learning models.

Explaining Models with Shapley Values

For our advances in improving explainability in yield prediction scenarios, we will extend the idea of Shapley values. Shapley values are a game-theoretic measure for solving a

fair distribution of resources in a cooperative game. The value awarded to a player is calculated by averaging his contribution over all possible coalitions that he could join within the game. The Shapley value was popularized as a feature importance measure by Lundberg & Lee (2017). The idea is to assign each feature an importance value for a particular prediction. The choice of the game-theoretic construct to solve cooperative games, namely the Shapley value, was found because of its desirable theoretical properties and results that are in line with human intuition. One of the mathematical properties allows them to provide an additive feature attribution method, which means that the sum of the feature importances will equal the actual model output for the example. The calculation of Shapley values is, in general, a NP-hard problem. But for decision trees, exploiting the tree structure allows computations in polynomial time, as explained by Lundberg et al. (2020), giving the first polynomial-time algorithm to compute explanations on tree structures based on game theory. The work also gives an idea of how to use many local explanations to represent the global structure of the model. Lastly, we want to highlight other efforts to extend the Shapley value feature importances towards groups of features. On the one hand, we have the classical game-theoretic view on this topic. However, the relevant works (Grabisch & Roubens 1999, Marichal et al. 2007, Flores et al. 2019) all fail to preserve the efficiency property, which means that the sum of all the attribution values of the features will not coincide with the output of the model and therefore are not suitable to base the explanations on. The work of Jullum et al. (2021) recognizes this weakness and presents a form to extend the Shapley value to groups of players in the context of feature importance. This allows for easier representation of the results together with a lower computational complexity. Amoukou et al. (2021) base an approach to evaluate groups of features on a different definition of grouped Shapley values, where groups of players continue to play against individuals. The work is extended by giving a fast computation for tree structures and selecting minimal subsets of features, so that the classifier will make the same decision with high probability.

After summarizing the landscape of general explainability methods above and the usage of Shapley values in this subsection, we can elaborate why we chose Shapley values as the basis for our explanations. First and foremost, Shapely values for explaining models are based on solid theory, with four axioms — efficiency, symmetry, dummy, and additivity — giving a theoretical foundation to the explanations. Most importantly, the axiom of efficiency will result in distributing the model prediction fairly on all features, such that the sum of all calculated importance scores together with the mean expected output of the model will precisely explain the model output for the data point at hand. Furthermore, in contradiction to one of the most used competitors in LIME, we need no assumptions like local linear behavior to explain the model with Shapley values. One of the biggest drawbacks of Shapley value explanations is usually the run time for arbitrary models, as we need to handle an exponential amount of coalitions during the calculation. As we focus primarily on random forest-based models within this thesis, this drawback does not apply as fast and exact calculation is available for random forests. Still, this makes our methods applicable and comparable for any machine learning algorithm, when we can omit runtime constraints or are satisfied with approximations of the Shapley value. Throughout this thesis, we will build on the existing work but focus on a definition of Shapley values where groups of players can only compete against other groups and

not individuals. Following this concept, we are the first to give a polynomial algorithm for the calculation in this scenario.

2.3 Feature Selection

To review related work on feature selection, we again divide the research into two areas. First, we will give a broad overview of feature selection in general, especially focusing on the model-driven wrapper methods and embedded methods for random forests. We then discuss the use of Shapley values for feature selection, as can be found in the literature up to this point. This related work section is based on our previous work (Huber & Steinhage 2024d).

Feature Selection in General

Feature selection, in general, is a very well-researched topic within the field of machine learning (Dhal & Azad 2022, Venkatesh & Anuradha 2019, Chandrashekar & Sahin 2014). Historically, approaches to feature selection can be divided into three different families (Huan & Hiroshi 1998).

1. Filter approaches, in which the feature subset is selected based on some quality measurement that is independent of any machine learning algorithm. An example of a filter method is a correlation-based feature selection, where features are greedily selected based on maximizing the correlation with the target variable while minimizing the correlation to already selected features (Li et al. 2017).

2. Wrapper methods, where feature selection is wrapped around a machine learning algorithm to generate a feature subset based on algorithm performance. In particular, (Huang et al. 2016) use a random forest to evaluate the value of a feature based on the performance of the model when the values of a feature are randomly permuted. The highest ranked features are then used for feature selection.

3. Embedded methods, where we directly compute the importance of the features based on their contribution to the machine learning algorithm. A famous example of embedded feature selection for random forests is a greedy feature selection based on the internal Gini importances of random forests (Menze et al. 2009).

Another related research direction is feature extraction, where the existing feature space is altered to create new more expressive representing features in a lower dimension. Although procedures like principal component analysis (PCA) (Park et al. 2005) are a very common paradigm in machine learning, we opt to focus on feature selection rather than feature extraction, as we lose explainability when we alter the feature space (Jijón-Palma et al. 2023).

Furthermore, as throughout our manuscript, we focus on problems where we already find a functional machine learning algorithm that should be further optimized, we will focus on wrapper methods and embedded methods for this literature review. As random forests have been used for several decades now, feature selection methods that wrap random forests or decision trees are a widely investigated research topic.

Deng & Runger (2012) are not selecting a subset of features before creating the model, but rather encourage the correct choice of features during model creation. This is implemented by penalizing the selection of new features within computing random forests when the feature information content is similar to a previously selected feature. Another wrapping method is proposed by Genuer et al. (2010). After creating a random forest, they evaluate the importance of features by calculating the difference in error when the feature values are randomly permuted. The deviation in error is then directly correlated with the importance of the feature, and the most important features are selected. This idea is further explored by the work of Gazzola & Jeong (2019) by adding a clustering of data points to take advantage of the structure of dependencies between the input features. The advantages of selecting features based on the knowledge already provided by the previously selected features are suggested by Alsahaf et al. (2022). Features are selected sequentially according to the internal importance of an XGBoost model, where in each step the probability of new features being selected is adjusted by the number of misclassified training examples for a model based on the current feature selection. A different approach is explored by Shih et al. (2018), which is not directly interested in finding a subset of features that are well suited to retraining a machine learning model with a smaller input feature space, and similar to our work, they focus on finding explanations based on individual data instances. The concept of sufficient reasons explains a prediction by finding a subset of features already sufficient such that the model output will be the same, no matter the other feature values. Arenas et al. (2022) extend this approach by not demanding the exact same model output, but a high probability of equality. Lastly, a wrapper approach is proposed for the selection of features in random forests by Zhou et al. (2021). They calculate feature weights dependent on the layers of the decision trees and select the feature with the largest weights as a partition for further model creation. The two feature selection approaches analyzed and developed in this thesis are all part of the family of wrapped feature selection for random forest. First, we use Shapley value feature importances that can be calculated for an existing machine learning model to select important features. This approach can also be seen as a heuristic search, as we use Shapely values as a heuristic on which features are expressive, without needing to consider every subset of features. Our other approach explained in this thesis will take the idea further, but employ it in an exhaustive search, where we evaluate every possible subset that can be selected.

Shapley Values for Feature Selection

With the rise of Shapley values in explainable AI, it also became an interesting research topic to explore the possibilities of Shapley values for feature selection. One of the main benefits of Shapley values in explainability is the ability to provide unified feature attribution values for different machine learning models. A similar benefit is given by the above-mentioned LIME framework. As with Shapley values, also LIME was researched as a feature selection tool (Man & Chan 2021), where in a comparison the top-rated features for both, a LIME and a Shapley values based feature selection improved the performance of random forests. One of the earliest works exploring Shapley values for feature selection is done by Cohen et al. (2005). After estimating the Shapley value feature importance via sampling permutations, they introduce the Contribution Selec-

tion Algorithm (CSA) to greedily select the most important features. They show the best results when using the algorithm to eliminate the unimportant features, instead of selecting the most important ones. They also investigated their CSA approach on additional data sets in follow-up work (Cohen et al. 2007). Marcílio & Eler (2020) give a general subsumption of the topic. The most important features according to TreeShap (Lundberg et al. 2020) are greedily selected and the accuracies reached are evaluated on several data sets. The survey shows the capabilities compared to other popular feature selection methods. A broad overview on the use of the Shapley value in machine learning by Rozemberczki et al. (2022) also talks about the possibilities of Shapley values in feature selection. Chu & Chan (2020) eliminate the problem of additional features altering the Shapley value and therefore the final selection of features. They do so by proposing an iterative feature selection approach based not only on Shapley values but also on higher-order interactions. We see multiple cases of Shapley value feature selection that solve real-world problems. Fang et al. (2022) successfully decrease the number of features needed for air pollution forecasting in China, by selecting the most important features of an ensemble model, including a random forest, according to the output of Shapley value-based SAGE explanations (Covert et al. 2020). Zacharias et al. (2022) designed a framework for easy access to a wrapped feature selection based on Shapley values and random forests. The result is a feature selection process that includes user feedback. They are able to report nearly unchanged accuracy for reduced feature counts on a variety of data sets. Strumbelj & Kononenko (2010) give a different procedure to obtain feature attributions with the Shapley value. As is commonly done, they randomly sample permutations to reduce computational complexity compared to calculating the exact Shapley value. The novelty of their approach is to obtain explanations by also randomly sampling an instance from the data that serves as the base value of the explanation in every iteration of the algorithm. Štrumbelj & Kononenko (2014) extend on this idea by improving the sampling algorithm via quasi-random and adaptive sampling, and are able to show improved explainability in an experiment with human participants. In comparison to those successful implementations of Shapley values for feature selection, we find some criticism within the literature (Kumar et al. 2020, Huang & Marques-Silva 2023, Sundararajan & Najmi 2020, Fryer et al. 2021). We will give an in-depth analysis of these critical ideas in Section 4.3.

All in all, we see that there exist multiple approaches to calculate feature attributions Shapley values and different ideas to take advantage of the calculated values for feature selection. Our work is the first to conceptually define the conditions that a definition of Shapley values should meet so that the resulting values can be used for feature selection. Furthermore, we give the first algorithm to evaluate the impact of the model averaging problem, where unimportant features can alter the final selection that is intrinsic to any definition of Shapley values in machine learning, and we give an extensive analysis on multiple real-world examples.

2.4 Filling Gaps in Remote Sensing LST Data

The following section is based on the Related Work section of our recent publication Huber et al. (2024b) and is again divided, covering two general research directions related

to the challenge of cloud-induced holes in remote sensing data. Remote sensing LST image interpolation is a widely investigated research area, often relying on statistical approaches to interpolate between close neighbors of missing values. This is covered in the first part of this section. The second part of the related work describes the state of general image inpainting with deep learning and why this can be related to providing gap-free estimations of LST.

Estimating Missing Land Surface Temperature

The common idea in most efforts to fill in gaps in LST data retrieved from remote sensing observations is to fit some kind of function to valid pixels neighboring the missing values, either in the spatial or temporal domain. An early adoption of this idea can be found in the work of Neteler (2010), which fills gaps in MODIS LST in mountainous environments. The study region is the central-eastern alps, and gap filling is done through statistical operations that rely on the valid nearest neighbors of missing data points. Finally, an evaluation against meteorological data measurements reveals R2 correlation values between 0.7 and 0.98. The author concludes that the new reconstructed LST time series are reducing the gap between high spatial and high temporal resolution. Metz et al. (2017) introduce an approach that reconstructs values in time through local weighted regression with a polynomial of order 2 and considering the 5 closest neighbors in time. Afterward, spatial imputation is performed with thin plate spline interpolation, including information on emissivity and elevation. All this is tested on MODIS data to achieve a gap-free coverage of central Europe, achieving remarkable results in dealing with extreme events, such as a heatwave effecting Europe in 2003. Similarly, Zhang et al. (2022) infer the temperature of the surface of the land using a two-step algorithm. The first step consists of data filtering and cleaning, where the missing value for gaps and low-quality pixels is computed utilizing the information from other available data from the same day, i.e., observations from other satellite overpasses. In the second step, the overall yearly temporal trend is used as a basis to calculate the missing residuals by fitting a smooth spline function to accessible valid pixels. They report an average Root Mean Squared Error (RMSE) of 1.88 °C, validated by filling artificial gaps of varying sizes. They highlight, that with their approach, there are no obvious block effects caused by large areas of missing values, which might exists in other LST data sets. Another popular approach is to transfer the LST data to a different domain to fit an interpolation function. Pham et al. (2019) fill gaps in MODIS LST data captured in Australia. They use multidimensional robust smooth regression (Garcia 2010) minimizing the squared error after performing a three-dimensional discrete cosine transform with a manually adjusted smoothing parameter. Validation is carried out against ground-based LST data on real-world gaps, revealing RMSE values between 2 °C and 3.9 °C. Similarly, Liu et al. (2020) use the same discrete cosine transform and penalization of least squares to fill gaps in the MODIS LST data. Synthetic gaps are produced by the random blinking algorithm to create evaluation data. An average RMSE of 0.91 °C is reported. Both works have the advantage of not relying on alternative geospatial data sets to fill the gaps and therefore avoid additional uncertainty. Furthermore, both works use a parameterization of the smoothing parameters in a way that retains the high-frequency components and therefore the global spatial patterns of the original LST data. A three-

step hybrid method is proposed by Li et al. (2018). They first used other satellite overpasses from the same day to fill gaps, if possible. This is followed by spatio-temporal gap filling and lastly temporal interpolation using neighboring days. The evaluation was carried out in urban areas within the US with an RMSE between 2.7 °C and 3.3 °C, using artificial masks of missing values taken from other real-world examples for the evaluation. Again, the approach does not rely on other geospatial data sets for gap-filling. An included daily merging step that uses other satellite overpasses of the same day further decreases the computational complexity of the problem. Focusing on the reconstruction of 8-day MODIS LST data, Xu & Shen (2013) used harmonic analysis of time series to reconstruct missing values. The validation was carried out by removing the information of 76 pixels and considering them as ground-truth data to calculate a mean absolute error of 1.51 °C. The authors highlight the capabilities of the approach to deal with large and persistent gaps in LST data, while struggling with sudden and extreme changes in the temperature. The first adoption of machine learning for the reconstruction of missing LST values can also be found in the literature. Li et al. (2021) produce a gap-free LST product by combining MODIS and ground-site measurements using random forests, achieving an RMSE of 2.756 °C when validated with data at the ground sites. Insights of their work include a clear improvement of the results for clear-sky conditions when compared with cloudy sky conditions and good performances even without the usage of spatial information for the reconstruction. Xiao et al. (2023) reconstructed LST MODIS data for Zhejiang province. Missing pixel values are calculated individually with different Machine Learning models, where eXtreme Gradient Boosting (XGBoost) outperformed the other approaches with an average R2 of 0.95. The authors conclude, that the XGBoost model demonstrated stronger learning capabilities and was able to fit the complex relations of the data, in comparison to statistical regression approaches or random forests. Both machine learning-focused works show the capabilities of tree-based approaches but have the drawback of needing to interpolate every missing value individually.

Overall, we see that the landscape of filling gaps in LST data focuses on statistical approaches and slowly begins to adapt machine learning methods into the process. A common drawback for all related work is the need to fill every missing value individually, either by fitting some kind of interpolating function or evaluating some kind of machine learning algorithm. This can be improved by our idea of using deep learning to solve the problem. With respect to the type of evaluation, the different approaches vary widely, making a direct comparison difficult. We analyzed the work of Metz et al. (2017) as an excellent execution of the statistical approach to interpolating LST data, and we will use this to obtain a baseline score for our use case and provide a fair evaluation.

Deep learning for Image Inpainting

Our idea of using deep learning to fill in gaps in MODIS LST data is highly influenced by general image painting problems. Inpainting of images, in general, refers to a problem in which information about the input image is missing in some way or another. The task is then to infer the missing information in a way that is most likely given the existing valid pixel information. It has been widely used in different applications to reconstruct image data. Many approaches focus on inpainting regular shapes, such as boxes in the center

of an image (Yang et al. 2017, Pathak et al. 2016). This inherently causes the network to be induced with bias that is usually not wanted in practice, and also something that, in general, rarely happens in the real-world. PatchMatch by Barnes et al. (2009) has long been the state-of-the-art method for image inpainting. PatchMatch suggests a random sampling base algorithm that computes patches that look similar to the holes to be filled in an image. However, the proposed algorithm only works well as long as there is a patch that is similar to the one already existing, as otherwise the results will quickly turn out to be visually unappealing and unnatural. Liu et al. (2018) introduced partial convolutions in 2018 to overcome the challenge of filling irregular holes. The method does not need to compute similarity metrics for patch-based inpainting, but rather just masks out invalid pixels during the convolutional step so that, rather than using wrong data, no data are used. After each convolution, the number of unfilled pixels decreases. The results have shown a huge improvement in the visual appearance, and it was one of the first papers to obtain good results for irregular holes. Furthermore, Han & Howe (2023) used 3D partial convolutions in the context of 3D histograms. Histograms depicted taxi pick-ups and bike sharing data in New York. They used a custom 3D variant of the U-Net introduced Ronneberger et al. (2015). One of the few papers using image inpainting for geoscience tasks is done by Sun et al. (2022). They implement a coarse-to-fine task-driven neural network that preserves good visual appearances but is also more suited to geoscience tasks where visual appearance is not as important as predicting specific values. Hence, a coarse model predicts the general appearance and a refinement net refines these predictions.

None of the previous work considered LST data as input for an image inpainting network. This leads to unsolved challenges, such as images consisting of only missing values that need to be interpolated. We solve this problem by including ground station data within our work and give proof that the patterns learned by deep learning architectures can help to produce precise inference of missing LST data by learning the inherent climate patterns of the interest region.

2.5 Knowledge Transfer for Yield Prediction

Lastly, we investigate the related work on the problem of reusing knowledge between different yield prediction domains based on a previous publication (Huber et al. 2024a). The related work that influences our research can be categorized into two main groups. First, we look at general developments in transfer learning, as it is mostly used for computer vision tasks. Second, we examine advances in transfer learning for remote sensing applications and yield prediction.

General Transfer Learning

Transfer learning is a topic of increasing popularity in different research areas. Most commonly, computer vision tasks are solved with the help of deep learning networks that are pre-trained on huge task-agnostic data sets, before being fine-tuned to solve specific problems. Plested & Gedeon (2022) give an in-depth survey of the state of transfer learning. Due to the versatile applicability of CNNs, the application of transfer learning methods to CNNs represents a well-known study objective. In image classification,

transfer learning methods achieve significant success (Sharif Razavian et al. 2014). In this work, we examine whether this success can be translated into yield prediction while following the insights of the community. Huh et al. (2016) found out, that a larger data set for pre-training results in better model performance. Furthermore, the number of layers that should be transferred during the training process depends on the similarity between the target data set and the source data set (Chu et al. 2016).

A common problem in transfer learning is the negative transfer of knowledge (Pan & Yang 2009). Negative transfer occurs when the source data set used for pre-training and the target data set are not well related, and transfer learning has a negative impact on the model accuracy. According to Plested & Gedeon (2022), a greater similarity between the transfer learning domains improves the ability to transfer knowledge between domains. If the domains are not well aligned, the key to overcome negative transfer is regularization, which restricts the amount of knowledge that can be lost during the fine-tuning step of transfer learning. The L^2-SP regularization method achieves success in dealing with negative transfer through an L^2 regularization with the origin parameters of the more general model (Xuhong et al. 2018). In our use case, this means that features that are extracted from the patterns of remote sensing data can be preserved during the transfer learning. Similar results can be achieved with DELTA regularization (Li et al. 2019), following the idea of only altering CNN channels that are not already useful for the target task. Batch Spectral Shrinkage (BSS) is another regularization method that often successfully eliminates negative transfer by suppressing non-transferable spectral components (Chen et al. 2019). Chen et al. (2019) report that BSS will never negatively affect performance in a given data set and is therefore also considered within our work to stabilize transfer learning.

Our work is the first to examine the extended use of regularization and transfer learning on hyperspectral remote sensing data presented as histograms, investigating how ideas developed for classical computer vision can help us with remote sensing applications.

Transfer Learning for Remote Sensing Applications

Transfer learning for remote sensing applications is a widely investigated topic. A recent extensive survey on transfer learning on remote sensing data is conducted by Ma et al. (2024). Most commonly, the task of land use and land cover classification is improved by starting from pre-trained models. Dastour & Hassan (2023) give an in-depth analysis of different deep learning architectures for this task. In general, all CNN tested are pre-trained on the well-known ImageNet data set (Deng et al. 2009) and applied to land cover classification on Sentinel-2A images. ResNet50 (He et al. 2016) was found to be the best architecture for transfer learning in this scenario. Li et al. (2020) show that a ResNet architecture again works best on the HSRRS data set for scene classification in urban populated areas, after being pre-trained on the ImageNet data set. Similar results are shown in the work of Alem & Kumar (2022) where a ResNet50 architecture pre-trained on ImageNet data shows high accuracy for land cover classification on remote sensing data. Tseng et al. (2022) use the pre-trained ResNet architecture inside Faster R-CNN as a backbone for rice seedling detection in RGB images outperforming a support vector machine. Chen et al. (2022) use the same pre-trained architecture, the Faster R-

CNN network. First, features are learned on the ImageNet data set before the network is fine-tuned to detect objects in high-resolution satellite images. Hilal et al. (2022) extend the idea of a pre-trained ResNet50 for land cover classification by including discrete local binary patterns to the ResNet features for the final classification. In very recent advances, Ma et al. (2023) improve the idea of Domain-Adversial Neural Networks (DANNs) (Ganin et al. 2016), where transfer learning is extended by projecting the input features of each domain into a common subspace. Their proposed partial DANN (PDANN) applies weights to the source samples according to their estimated yield distribution in the target domain and is used to improve the transfer learning of soybean and corn yields between different regions in the US. Lastly, as mentioned above Wang et al. (2018) first train an LSTM network for soybean yield prediction in Argentina and then transfer the knowledge to regions in Brazil. Khaki et al. (2021) combine the predictions of different crops and present a multi-target regression CNN structure, simultaneously predicting soybean and corn yields in the US based on remote sensing data.

Our work is quite distinguishable from the approaches described above. As our input data are processed to be represented by histograms and include multiple hyperspectral image channels and our target is not a classification but a regression, it is not possible to use huge image data sets like ImageNet as a source domain for transfer learning. Even in our source domain, we have relatively small amounts of training data compared to modern image classification tasks. We need to overcome this issue by using deliberate regularization methods during knowledge transfer.

3 Material

In this thesis, we use two data sets to test and evaluate our methods. First, we describe an international data set for the prediction of soybean yields. The data captured by the USDA are one of the largest data sets available for yield prediction, and reporting results enables us to compare our work with the state-of-the-art for yield prediction. The evaluation of the methods in a scientific context is done on these data in Chapter 5. Second, we give an in-depth description of the data used within the KI-iREPro project. These are the data that motivated the research of our approaches and will be used to evaluate the reconstruction of remote sensing data in Section 5.4 and the real-world application of our methods in Chapter 6. Lastly, we introduce the changes made to the soybean data set presented in Section 3.1 and the additional data on the prediction of soybean yields in Argentina, used to transfer knowledge between the two domains.

3.1 Soybean Yield Prediction - An International Data Set

For this section, we directly follow the presentation of the data as previously published (Huber et al. 2022). The data chosen to evaluate different yield prediction approaches are soybean production reported within the US over a period of time of 2003 to 2021. This allows for good comparability with previous work, as soybeans in the US are widely investigated (You et al. 2017, Sun et al. 2019) and form one of the largest databases for yield prediction tasks. The study area includes 13 adjacent states within the US. The states are Arkansas, Illinois, Indiana, Iowa, Kansas, Minnesota, Mississippi, Missouri, Nebraska, North Dakota, Ohio, South Dakota, and Wisconsin. The 13 states are highlighted in Figure 3.1, showing that most of the blue highlighted soybean farmland in the US is located within the selected states. The study spans 19 years, giving us 14543 data points in total. For the end-of-year soybean yield prediction, predictions are based on data captured over the time span between the 49th and 321st days of the year. Since the soybean cycle varies between years and state selection, generous ranges are used around the usual dates for planting and harvesting. The range includes approximately two months before the first crops are planted and ends approximately one week after the last soybeans are harvested (USDA 2010). The input features for the prediction of the yield are extracted from five different data sources that have also been used to predict the soybean yield by Sun et al. (2019).

USDA Yield Data

County-level soybean yield data from 2003 to 2021 are collected by the USDA (USDA 2021). Soybean yields are reported in soybean bushels per acre (bu/acre), with 1 bu/acre approximately equivalent to 67.26 kg/ha. The yield data are used as ground-truth labels for model training and validation. Figure 3.2 shows the historical yields during the study

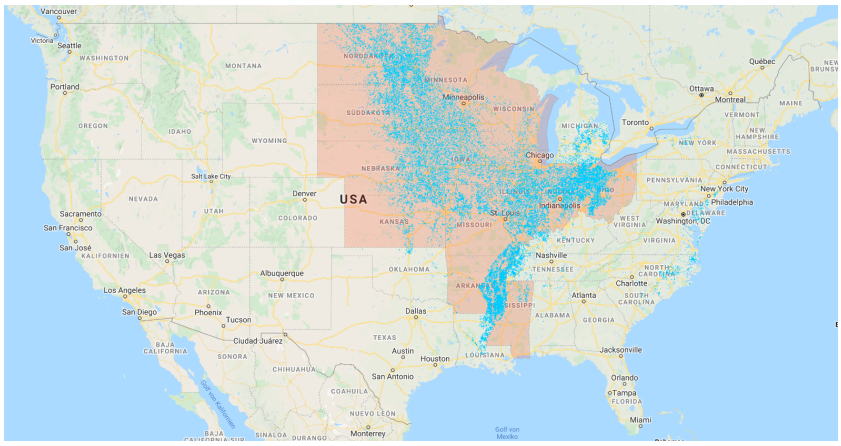


Figure 3.1: The 13 states of the US used to extract historical yield data highlighted in red. The blue areas indicate soybean farmland. The image is extracted via Google Earth Engine (Gorelick et al. 2017) and previously published (Huber et al. 2022).

period, and the graph shows an overall upward trend, caused mainly by technological innovations that lead to more efficient farming (Chambers & Pieralli 2020).

MODIS Surface Reflectance and Temperature

Within this study, remote sensing data collected by Moderate Resolution Imaging Spectroradiometers (MODIS) installed on the NASA Terra and Aqua satellites are used. MODIS data are available from the Google Earth Engine Catalog (Gorelick et al. 2017) and consist of MODIS products MOD09A1 that provide surface spectral reflectance data and MYD11A2 that provide temperature data for the surface of the land. The surface reflections of light are captured in a range starting from the visible spectrum (459-479 nm) to the invisible infrared spectrum (2105-2155 nm) and are captured by 7 different channels. Each channel is referred to as a band in the context of remote sensing images. In this study, both products are used at a resolution of 500 m per pixel, with each image being an 8-day composite of the values of each band. The visible spectrum includes blue, green, and red light, which can be used, e.g., for RGB visualizations of the regions. The non-visible spectrums are four different wavelengths of infrared light and near-infrared light. The importance of near-infrared light for yield prediction is well known and is used in standard measures such as the Normalized Difference Vegetation Index (NDVI) (Gao 1996). A temperature image shows two bands for the temperature during the day and night. Using the timespan between the 49th and 321st days of the year therefore produces 34 images to correspond to one year's yields.

Daymet Weather Data: Precipitation and Vapor Pressure

The Daymet V4 data set offers daily surface weather and climatological summaries extracted from the Google Earth Engine Catalog (Thornton et al. 2016). It provides gridded estimates of daily weather parameters based on selected meteorological station data and various supporting data sources within the US. Two important weather param-

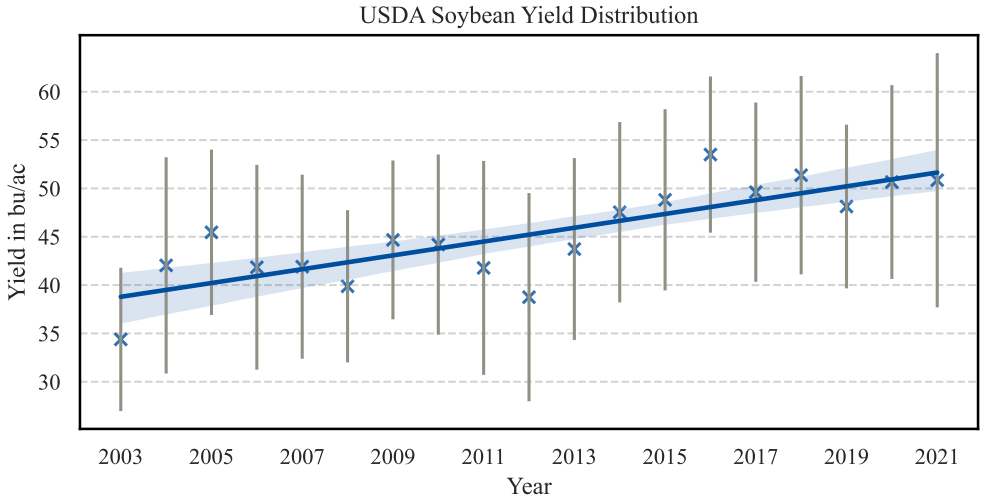


Figure 3.2: Historic soybean yields in the US. The crosses indicate the mean yields in bu/ac with the grey bars describing the standard deviations. The blue line shows the rising trend in annual yields, together with a confidence interval.

eters in Daymet, daily total precipitation and daily averaged partial pressure of water vapor — produced on a 1 km × 1 km gridded surface over North America were selected as climatic factors as proposed by Sun et al. (2019). The scaling is changed from 1000 m to 500 m using the native Earth Engine image pyramid to match the weather data with the other yield and environmental data. Lastly, the daily data are aggregated to match the same 8-day composites of the MODIS data by applying the mean.

Tiger County Borders

The TIGER data set from the US Census Bureau contains the 2018 boundaries for the primary legal divisions of the US states (Bureau 2018). Information is used to crop remote sensing data to county borders.

USDA NASS Cropland Data Layer

The Cropland Data Layer (CDL) is a crop-specific land cover data layer created annually for the continental US and also downloaded from the Google Earth Engine catalog. The selection of 13 adjacent states within the US has been partially covered since 2003 and fully covered since 2006. The states of Kansas, Mississippi, Missouri, Ohio, and South Dakota have also been covered since 2006, when the CDL started to be available to the entire US. Originally, the CDL has a resolution of 30 m. This resolution of 30 m is upscaled to a resolution of 500 m to match the resolutions of the other data sources explained above. The CDL information is used to focus the training of the prediction system to the parts of the remote sensing images that cover soybean farmland.

3.2 Grapevine Yield Prediction in KI-iREPro

The KI-iREPro project aims to predict grapevine yields in Rhineland-Palatinate, Germany. For this task, we have two prediction approaches, (1) at the plot level for commercial use and (2) at the plant level for help in developing new varieties. The data set for these tasks varies in two main directions compared to the soybean yield data described above. The first distinction comes from the monitored crop. Although soybeans are planted and harvested in an annual cycle, grapevines are not destroyed during harvest, and the same plant will carry fruit over decades. Furthermore, soybean yields are optimized with respect to quantity, as the quality is very stable. However, for grapevines, quality is the main factor influencing winemakers' profits, and the quantity of the yields is regulated by federal laws. The second distinction between the two data sets comes from the granularity of the data. For the soybean yield prediction, we always cover whole counties with each prediction, making the appearance of cloud induced holes in remote sensing images less threatening to the data, as it is unusual for such a large area to be covered in clouds for multiple days. For the grapevine yield prediction, we need the data to cover individual parcels, each only a few square kilometers big. This corresponds to only a handful of pixels in remote sensing satellite images, which creates the need to fill the satellite data with interpolation processes. Furthermore, this allows us to use only a certain kind of satellite data, the Land Surface Temperature (LST), since it is the only one where holes can be filled with the help of ground-site weather stations and where the distribution is smooth enough to allow for interpolation. The descriptions in this section of the study area and the LST data used are based on our previous research (Huber et al. 2024b) and adjusted to fit the scope of this thesis.

Study Area

The study area of the KI-iREPro project is located in Rhineland-Palatinate, Germany. The area measures 47.49 km by 36.96 km and is centered around 8.112 °N and 49.25 °W. It includes vineyards that should be monitored throughout the year. An overview of the location in the south west of Germany can be seen in Figure 3.3 a), where we see that it is located south of Frankfurt am Main and north west of Stuttgart. To improve the interpolation of missing values inside the study area, we added a 10 km padding around the relevant area, which will not be considered during evaluations. In Figure 3.3 the original research area is represented by a purple rectangle, while the red rectangle shows the extended area that includes the padding. The study time frame for LST interpolation is set from 11.03.2008 to 15.11.2022, as dictated by the availability of ground-site weather station air temperature data and the availability of remote sensing data at the time of our study. The relative location of the weather stations is shown in Figure 3.3 b). The region itself offers some interesting topology, with heights ranging from 91 to 599 m above zero, including a steep change in elevation along an axis from the south west to the north east corner, as shown in Figure 3.3 c).

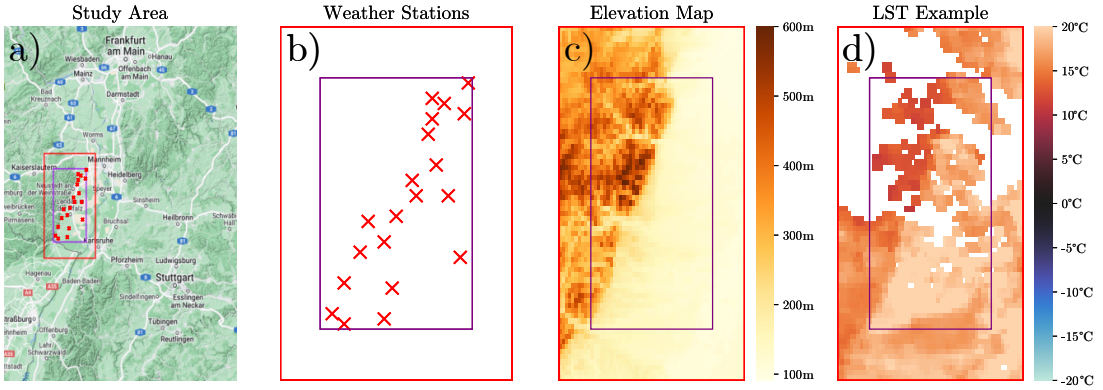


Figure 3.3: Overview of the KI-iREPro study area, weather stations, elevation, and occluded LST data. a) The study area is located mostly in Rhineland-Palatinate, Germany, measuring 47.49 km by 36.96 km, with the extended padding shown as a red rectangle and the weather stations shown as red crosses. b) The relative location of the weather stations. c) Elevation map showing a characteristic pattern. d) An example for LST data on 02.09.2010 showing gaps in the data and the characteristic temperature distribution dictated by the elevation. Figure previously published (Huber et al. 2024b).

MODIS Data and Occlusions

Our source for remote sensing LST images is the well-known MOD11 product (Wan et al. 2023), as it is also used for the soybean yield prediction data. The LST data are retrieved from a multitude of inputs from the MODIS sensor attached to the Terra satellite provided by NASA using the generalized split window algorithm (Wan & Dozier 1996). In summary, the algorithm consists of three steps. First, cloudy pixels are detected and skipped in LST production. Second, the estimation of atmospheric column water vapor and lower boundary temperature from seasonal climatological data to improve the accuracy of LST. Third, the estimation of band emissivities correcting for atmospheric effects at specific wavelengths that are used to finally retrieve the LST information. Other existing data sources for LST records include the Landsat collection that provides LST records at a high spatial resolution but lacks the daily coverage of MODIS that is important for a variety of tasks.

In our case, the data are downloaded using the Google Earth engine (Gorelick et al. 2017). The images were taken with a spatial resolution of 1 km. An example of a partial LST image recovered from the MODIS sensor is shown in Figure 3.3 d). To have as much usable data as possible, we did not consider the quality indicator to remove any further values, as early experiments indicate that, in general, every pixel with a value attached can be considered good enough to be used for further processing. Together with the LST, the product provides information on the emissivity of each pixel. The emissivity is considered when evaluating statistical approaches for gap filling but did not show any added benefit for our approach. The satellite regularly observes our study region around 11:00 am, monitoring temperatures from -32 °C to 46 °C with a mean temperature of 12.69 °C, a median temperature of 12.59 °C and a standard deviation of 10.3 °C when considering all valid pixel values in our research area. When considering all 5706 observations in our time frame, 33.24% of all pixels are filled with valid information.

This includes a total of 280 images where all 4012 pixels are valid and 1500 images where no pixels are valid. On average, each image consists of about 1507 valid pixels that can be used to estimate the values of the remaining 2505 missing pixels.

After interpolation, we use the data as basis to match the climatic conditions with the ground-truth grapevine yields. To use these data for yield prediction, we start by matching each parcel with a pixel of the interpolated LST image, providing daily records of the LST for the parcel. To obtain a feature-based representation of the data, we aggregated the LST data in 35 timeframes, each covering 10 days, going backwards from the known date of harvest. This ensures to cover almost the whole period of plant development, which is necessary for grapevines as a perennial plant. For each 10-day window, we calculate the average temperature as a representation, and for all 10-day windows, we calculate the biggest positive and negative difference in temperature observed within one window. Furthermore, we calculate an array of aggregated statistics over the entire 350 days. We counted the number of days when the LST at the time of the capture was below 0, -2, -4 and -6 °C. Similarly, we counted the days when the temperature was above 10, 20, 30, 35 and 40 °C. We count the days between the last temperature below zero and the harvest date, such as the temperature sum of all 350 daily records.

Ground Site Air Temperature

To be able to fill the gaps in the LST data, even when the entire padded study area is occluded, we decided to include air temperature measured by local ground-site weather stations in our pipeline. Air temperature and LST are of course different measurements with varying records up to multiple °C, but often show a high correlation (Cao et al. 2021). A local ground-site weather station cannot monitor the varying climate in an extended region, but excels at providing precise and reliable information about a specific location. Using both weather stations and remote sensing data allows us to combine the best characteristics of both data sources, with the aim of providing reliable LST information with high resolution. We use 20 weather stations in our study area with their locations shown in Figure 3.3 b), located mainly in the low-lying regions of our research area. Weather stations can capture a multitude of features at an hourly rate. We are mostly interested in the data captured at the time when the MODIS satellite observes the region, together with the data a few hours earlier, because both can give information on how the measured air temperature can be homogenized with the surface temperature that is of interest to our study. We use the average, minimum, and maximum air temperature, together with the air humidity captured at 6:00 am. and 11:00 am., to infer the surface temperature at 11:00 am. clock of the pertaining day. All features are measured twice, once at a height of 20 cm and once at a height of 200 cm. A list of all weather stations and their coordinates is given in Table 4.1. Data are captured on account of the federal state Rhineland-Palatinate, Germany (Agrarmeteorologie RLP 2023).

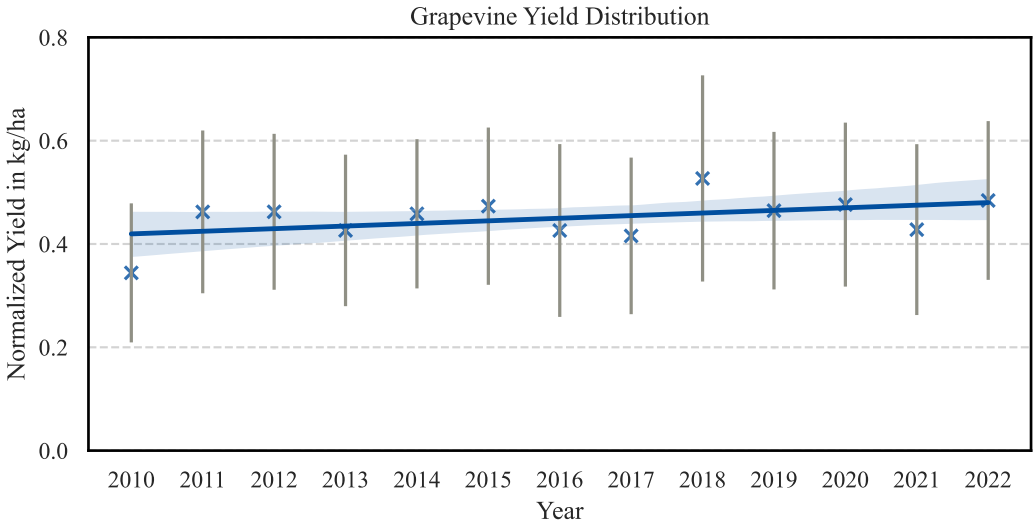


Figure 3.4: Normalized grapevine yields captured within the KI-iREPro project showing the annual differences in the harvested yields. The cross shows the mean value, the gray bar shows the standard deviation and in blue we see the trendline.

Grapevine Yield Data

We obtain ground-truth yield data from the delivery notes of a winemaking collective located in Rhineland-Palatinate, Germany. In a winemaking collective, vineyard owners and winemakers deliver grapes to the cooperative, which is then in turn involved in the production of wine and the subsequent marketing. Therefore, our data set involves vineyards of 396 different winemakers. Our primary data source are the delivery notes collected from the collective between 2010 and 2022. The entire data set contains 11624 data points, averaging approximately 894 data points a year. As general processing in viticulture is not necessarily made with singular parcels in mind, we need to clarify that one data point contains the aggregated yields of one winemaker in a year with respect to a specific variety. This is necessary to secure the accuracy of our ground-truth yield data given the agricultural practices in viticulture. From the delivery notes, we are already able to extract multiple crucial features involved in the modeling. First, we can extract ground-truth yields in kg/m^2 that will serve as a target variable during model training and evaluation. To anonymize the confidential data, we normalized the yield data to the 0 to 1 interval by using the biggest and smallest yield records within the training data. The average normalized yields over the years are shown in Figure 3.4. Other information that we can extract from the delivery notes that will be useful for modeling includes the variety, the year the vineyard was planted, an anonymized ID of the winemaker, and the harvest date. The variety contains important information, as different varieties produce different amounts of yield. Our data set contains 38 different varieties with differing amounts of data points, ranging from Dornfelder with 1558 different data points to Frühburgunder with only one data point within our data set. The year the vineyard was planted can offer information about the capacity of the plants at hand, as grapevines as a perennial plant show certain behavior depending on their age. Here, our data show

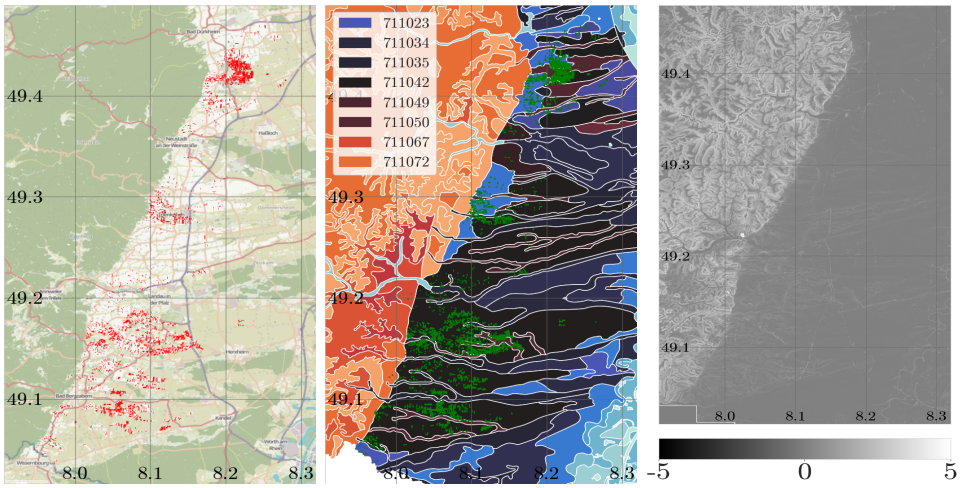
different vineyards, varying between being just planted and being present for 75 years. The anonymized winemaker ID serves as a substitute for the missing information about in-field processed that is not captured within our data. The idea here is that each of the 396 winemakers has some kind of preferences during the care of his plants, and therefore we allow the model to indirectly model those in the prediction. Lastly, the harvest date will be used to match the time series of climatic information with the data points for the predictions.

Elevation and Soil Data

To incorporate the elevation and inclination of the vineyards, we use a digital elevation model (DEM) with a resolution of 5 m. For each parcel, we used the mean values of latitude and longitude to place the parcel within our DEM and extract the elevation. The lowest parcel within our data lies on average at 75.75 m above sea level, with the highest parcel laying on average 289.65 m above sea level. To extract the inclination, we use the 3×3 Sobel operator (Sobel 2014). Originally designed to extract edges in images, it can be used to approximate the gradient in the x- and y-directions in an image, where the change in brightness in a grayscale image is analyzed. In our use case, the image consists of the DEM information, and the approximated gradient directly reveals the slope and inclination of each pixel in the image. The absolute value of the gradient averaged per field varies between 0.023 and 1.538, as shown in Figure 3.5 c) and the slope takes values from 0.186° to 12.271°. In addition, the average responses of the Sobel operator are used to determine the orientation of the inclination of the fields as a value between $-\pi$ and π, showing the possible rotation of 360°. Our source of soil information is the soil map of Germany 1:250,000 (BUEK250) which provides information on land use and soil in the states of Europe (Krug 2018). We used this information to one-hot encode the 6 different soil types in a categorical fashion. The soil map can be seen in Figure 3.5 b).

Grapevine Yield Prediction on Plant Level

Besides grapevine yield prediction for plot-level predictions, we work on plant-level predictions. For this, we use different phenological traits of individual grapevines as input to our machine learning models. The final goal of this task is to use the automatically generated data that are extracted from camera images taken within the vineyards. As this pipeline is still in development with our project partners within KI-iREPro, we use data acquired through manual plant appraisal to evaluate the concept. The data description in this section is based on one of our previous publications (Huber et al. 2024c). Data were captured in two experimental vineyard plots at the JKI institute for grapevine breeding Geilweilerhof located in Siebeldingen, Germany (49°13'07.0"N 8°02'45.0"E) in the year 2021. The rows were planted in the north south direction and the vines were grown in a vertical shoot positioned trellis system with a cane and around 10 buds per vine for both plots. The data set includes information on four well-established grape varieties, namely Dornfelder, Pinot noir, Pinot blanc, and Riesling, as well as seven elite breeding lines from the intermediate testing phase (Töpfer & Trapp 2022). The plants are grafted on SO4 root stocks with an interrow distance of 2 m and grapevine spacing



(a) Location of the individual vineyards (red).

(b) Different soil types shown in different colors.

(c) Sobel operator showing gradients.

Figure 3.5: The location of our test vineyards in the study region. The soil types in b) are used as a categorical feature during the prediction. The Sobel operator c) helps to identify the hillside location of the vineyards.

of 1.1 m for both plots. A complete overview of the grapevines used for data acquisition can be found in Table 3.1. As already explained, we used the data acquired from manual plant observations. Viticulture experts made optical observations seven times during the growing season. This leads to some features being present multiple times, which shows the development of the plant. The number of shoots, for example, is captured four times, as it is an integral feature when measuring the grapevine. The yields of the 400 grapevines are manually weighted to serve as the ground-truth for our predictions. Table 3.2 shows the dates on which the features are extracted, together with a description of each feature. For the predictions, we need to remove 30 data points due to missing values. The remaining data points have an average yield of 1.34 kg per grapevine.

Variety	VIVC	Accession number	Rows	Count	Year	Abb.
Dornfelder	3659	DEU098-2008-057	1	21	2008	Do
Pinot noir	9279	DEU098-2008-075	1	22	2008	PN
Pinot blanc	9272	DEU098-2008-072	1	22	2008	PB
Riesling	10077	DEU098-2008-080	1	24	2008	Ri
Gf.2010-011-0048	-	-	2	50	2015	BL1
Gf.2001-041-0004	-	-	2	46	2016	BL2
Gf.2001-041-0003	-	-	2	46	2016	BL3
Gf.2004-043-0010	-	-	2	46	2016	BL4
Gf.2004-043-0021	-	-	2	45	2016	BL5
Gf.2004-043-0034	-	-	2	40	2018	BL6
Gf.2000-305-0081	-	-	2	38	2019	BL7

Table 3.1: Overview of the different grapevines used for the plant level yield prediction. The data set consists of four established and seven experimental varieties. Table taken from Huber et al. (2024c)

Feature	Capturing Dates (DD.MM.)	Data Range
Number of shoots	03.05., 12.05., 02.06., 20.07.	1 - 22
Number of shoots with inflorescences	02.06.	1 - 22
Average inflorescences per shoot	02.06.	0.23 - 3.2
Number of inflorescences per vine	02.06., 16.06.	1 - 44
Number of shoots with bunches	20.07.	1 - 22
Average bunches per shoot	20.07.	1 - 3.9
Number of bunches per vine	01.07., 07.07., 20.07.	1 - 71

Table 3.2: Overview of the features extracted via manual plant appraisal together with the date the appraisal has taken place in 2021. Some features are captured multiple times during the study and at each appraisal multiple features are captured. Table taken from Huber et al. (2024c).

3.3 Unified Soybean Yield Prediction for Transfer Learning

To test our approaches to transfer knowledge between different domains for yield prediction, we introduce a second scenario for soybean yield prediction, this time in Argentina. Argentina is one of the main soybean producers in South America. However, the amount of available data differs compared to the US, with the country being much smaller and data acquisition less proficient. Therefore, Argentina is the perfect target domain to transfer the knowledge gained for the prediction of soybean yield in the US. We need to alter the data set described in Section 3.1 since we want to create compatibility between the two data sources. This is necessary, as, for example, the Daymet V4 climate data are only available in the US and not in Argentina. Furthermore, we adjust the time frame to cover 10 instead of 19 years, to retain maximum relevance of the training data for the actual use case, considering the rising trend for annual yields in both data sets. In addition, the selection of states is slightly different, focusing on the largest producers, as we lose access to a precise location of soybean farmland when using data in different countries than in the US. The use of less precise general farmland indicators forces us to only consider the counties with the highest percentage of farmland being soybean farmland to reduce noise in the data.

Yield Data

As mentioned above, we make slight changes to the selection of US data processed compared to Section 3.1. Soybean yield data are again taken from the US Department of Agriculture (USDA) for the US during the period 2010 to 2020, inclusive (USDA 2021). In Argentina, the yield data are taken from the Ministerio de Agricultura (2023) over the 2010/11 to 2020/2021 seasons, inclusive. This leaves us with a total of 8465 data points in the US and 1542 data points in Argentina. An overview of the yield data can be seen in Figure 3.6. Generally, we see that the yield data in the US and Argentina show similar patterns, with the relative yields in Argentina consistently lower than in the US. Both data sets show a positive linear trend that indicates more soybean yields in recent history. This is mainly due to technological innovations that lead to more efficient agriculture (Chambers & Pieralli 2020). Greater similarity between the domains of transfer learning improves the ability to transfer knowledge between domains. Here, the data for the US denote the source domain and the data for Argentina denote the

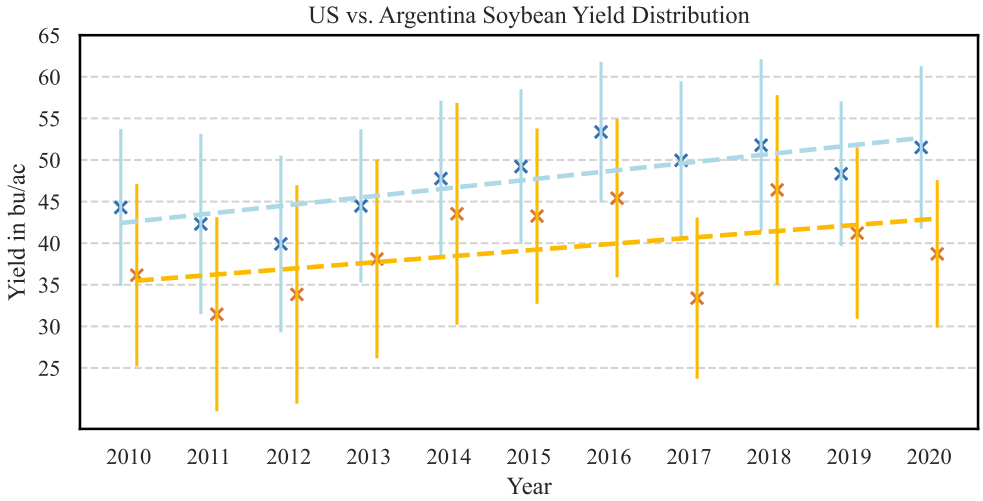


Figure 3.6: Visualization of the ground-truth yield data in the US (blue) and Argentina (orange). Average yields are indicated by the crosses and standard deviation are shown by the vertical lines, both in bushels per acre. The dashed lines indicate the linear trend in the data. We see the US yields to be generally higher than in Argentina.

target domain. In order to keep the similarity of the data as high as possible, the same data cleansing procedure is used for both domains. In our case, data cleansing serves to ensure a sufficiently large input data set. First, we select the regions of each country where soybeans are grown. This is the north eastern part of the US and the north of Argentina. Then, on the one hand, data points that cannot be assigned to a county due to missing information and counties with no yield specified or with a yield per hectare of zero are removed. On the other hand, all counties that have less than 2000 pixels of crop mask are removed. This is necessary to ensure a significant amount of soybean cropland for every data point and to reduce the number of noise from other sources.

Remote Sensing Data

For both data sets, we use a selection of remote sensing data very similar to the one already explained in Section 3.1. This includes surface reflections taken from MOD09A1 as an 8-day composite of the images taken by the satellites with a resolution of 500 m by 500 m. Climatic conditions are exclusively monitored by the MYD11A2 data. The original solution of 1 km by 1 km is resampled to match the resolution of 500 m by 500 m of the surface reflection data. We forego the use of Daymet data as they are exclusive to the US. Another difference is the quality of the mask used to filter out pixels that are not related to crop growth. For transfer learning, we use the MCD12Q1 data set (Friedl, M., Sulla-Menashe, D. 2019), which offers a yearly classification of several types of land cover with a resolution of 500 m by 500 m. Areas consisting of more than 60% cultivated fields are classified as croplands and will be the focus of our study. The more precise CDL used in Section 3.1 is again exclusive to the US and would not allow a knowledge transfer. Lastly, we crop the satellite images for the individual counties. Therefore, we require district assignments labeled with the county names. In the US, the same

TIGER data set from the US Census Bureau (Bureau 2018) is used for this purpose. In Argentina, we use the county (Departamentos in Argentina) mapping used by Wang et al. (2018), which contains all counties relevant to soybean cultivation.

4 Methods

The Methods chapter has five sections that match the five challenges and contributions defined in Section 1.1 and Section 1.2. We start by showing a pipeline to enable the usage of XGBoost for yield prediction in Section 4.1. We do so by modeling the underlying distributions of the remote sensing data, allowing us to train an XGBoost model, resulting in a structure similar to a random forest. To subsequently explain the model, we introduce the paradigm of Grouped Shapley Values (GSV), where we extend the popular Shapley values to obtain explanations of machine learning models for groups of features, as we often find them in yield prediction scenarios. This will be done in Section 4.2 together with the introduction of an algorithm to calculate the GSV in polynomial time for random forests. Section 4.3 is dedicated to the question if Shapley value feature attributions can be used for feature selection after a successful model is trained, i.e., reducing the dimension of the input feature space while preserving the accuracy of the modeling. We guide the discussion by first defining four necessary conditions to defining a (grouped) Shapley value for feature attributions that allows feature selection, and show that our definition used throughout this thesis fulfills all of them. We then approach flaws in the Shapley value feature selection tied to the nature of the approach to perform model averaging by introducing the concept of Conditional Feature Selection (CFS), allowing us to give an application-driven quantitative analysis of the impact of the problem. In the following two sections, we first show a solution to interpolate gaps in remote sensing LST data in Section 4.4. This is necessary to enable our methods established in the previous sections to be applied to yield prediction with a high spatio-temporal resolution, for example, when predicting the yields for individual plots. We do so by giving a U-Net deep learning architecture including partial convolutions and establishing a training loop that allows us to use exclusively partially available ground-truth data for training the model. Lastly, we conclude the chapter by describing the methods used to align two different yield prediction domains for transfer learning. As random forests are not well suited for transfer learning, when the domains for transfer learning are inhomogeneous, we present a deep learning architecture together with transfer learning techniques, including Batch Spectral Shrinking (BSS), L^2-SP regularization, and a Gaussian process, to improve the knowledge transfer.

4.1 A Yield Prediction Pipeline with Extreme Gradient Boosting

As mentioned in the Introduction, the content of this section is already published (Huber et al. 2022). We will describe how to enable the use of remote sensing satellite images for yield prediction with machine learning algorithms that need a tabular description of the data on the example of soybean yield prediction in the US. Following this, we

describe how state-of-the-art deep learning approaches address this task to prepare for the extensive evaluation in Section 5.1.

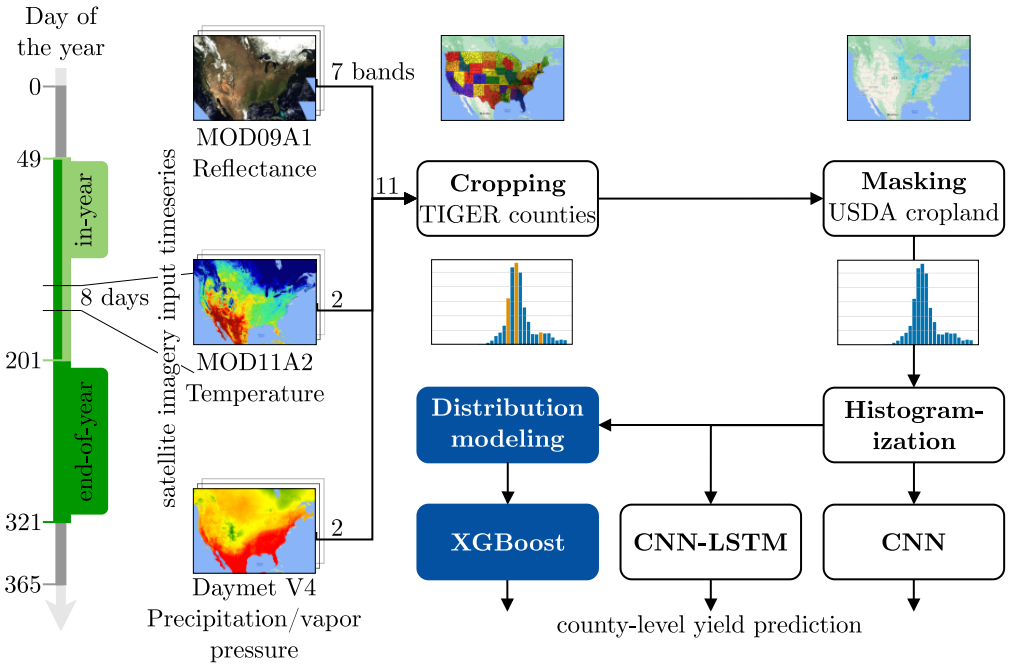


Figure 4.1: Overview of the remote sensing data processing pipeline for yield prediction. The aggregated remote sensing data are cropped and masked, before they are processed. The resulting data are then either represented by histograms for the deep learning approaches or a feature-based approximation of the underlying distribution for XGBoost. Figure taken and altered from Huber et al. (2022).

A graphical overview of the entire prediction pipeline is given in Figure 4.1. The process starts with determining the time window that is used for the predictions. We use two different windows to test our prediction model, first for the more difficult task of predicting the yield weeks before the harvest and a second one, where all harvests have already concluded. For the in-year prediction, we use data captured between day 49 and day 201 of the year. For the end-of-year prediction, the time frame is between day 49 and day 321 of the year. As explained above in Section 3.1 three different satellite products are utilized to build the input data for the pipeline. That is, the MOD09A1 reflectance data, the MOD11A2 temperature data, and the DaymetV4 precipitation and vapor pressure data. The relevant sections for the prediction of yield at the county level are obtained using the Tiger county borders to crop the data and by applying the USDA cropland data as a mask to keep only pixels covering soybean farmland. The experimental setup focuses on a direct comparison between XGBoost-based yield prediction and current state-of-the-art deep learning frameworks. After the cropping and masking explained above, the resulting images have irregular shapes and varying sizes between different counties. Furthermore, the preferred presentation of data as input for deep learning frameworks and classical machine learning approaches differs. Therefore, the data are processed towards either a histogram-based approach for deep learning or a feature-based modeling of the underlying pixel distribution for the XGBoost approach.

The histogramization will output the relative frequency of the pixel values in 32 evenly spaced intervals, and the distribution modeling will highlight three key characteristics of the underlying distribution. Lastly, the different approaches are applied on their respective preprocessed data sources to compute both the in-year and the end-of-year predictions.

Yield Prediction with Gradient Boosting

Among the machine learning methods used in practice, gradient tree boosting (GTB) or gradient boosted regression tree (GBRT) and especially, the XGBoost approach of Chen & Guestrin (2016) has been shown to give state-of-the-art results in many standard classification and regression benchmarks and competitions. For example, in KDDCup 2015 (Fournier-Viger 2016), all teams in the top 10 used XGBoost. A single regression tree is a set of cascading questions. Applied to a data point, i.e., a set of features and their values, the feature values are used to answer the cascading questions, yielding a result value. Regression trees are trained on training data to learn the appropriate cascade of questions. A tree ensemble is a family of K regression trees where the final prediction is the sum of the predictions for each tree. XGBoost learns the appropriate tree ensemble from training data by iteratively adding new trees into the given ensemble that maximize the prediction performance of the trees already in the ensemble plus the new tree that has to be added. In other words: for a given tree ensemble E_k of k trees and its residuals, a new tree e_{k+1} that fits best to these residuals is combined with the given ensemble E_k, which produces the boosted version E_{k+1} of E_k. The performance of E_{k+1} will be better than that of E_k. This can be done for K ($1 \leq k \leq K$) iterations, until the residuals have been minimized as much as possible. Therefore, the design of the new tree uses the gradients of the performance evaluation functions to select the best cascade of questions.

To extend the general explanation of the XGBoost algorithm, a more in-depth explanation of the XGBoost regression design is given below. In general, the following objective function needs to be optimized:

$$O(M) = L(M) + \Omega(M), \tag{4.1}$$

where M is the model learned from the training data. L is a differentiable convex training loss function, such as the Mean Squared Error (MSE) and Ω is the regularization term which prevents overfitting, i.e., the model learning the structure of the training data too precisely, leaving it without the ability to generalize to unseen data.

As explained above, the model is an ensemble of decision trees, so the output $\widehat{y}_i$ for each instance i is given by a collection E_K of K trees.

$$\widehat{y}_i = \sum_{k=1}^{K} e_k\left(x_i\right), e_k \in E_K. \tag{4.2}$$

Since the model in the general objective (4.1) includes the decision trees as functional parameters, the loss cannot be optimized directly, but it has to be done iteratively, where

in the t-th step the following is minimized:

$$O^{(t)} = \sum_{i=1}^{n} L\left(y_i, \widehat{y}_i^{(t)}\right) + \sum_{k=1}^{t} \Omega\left(e_k\right), \tag{4.3}$$

where n is the number of training data points, and for a given iteration, $\widehat{y}_i^{(t)}$ can be calculated as:

$$\widehat{y}_i^{(t)} = \sum_{k=1}^{t} e_k\left(x_i\right) = \widehat{y}_i^{(t-1)} + e_t\left(x_i\right), \tag{4.4}$$

by adding the prediction of the latest calculated tree to the previous prediction.

Chen & Guestrin (2016) define the regularization term $\Omega\left(e_k\right)$ as:

$$\Omega\left(e_k\right) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2, \tag{4.5}$$

Where T is the number of leaves in the tree penalized by γ and w_j are the leaf weights within the tree penalized by the parameter λ. The leaf weights represent the answer of the tree that is given for the regression task, when the cascade of questions reaches the particular leaf. Lower numbers of leaves with smaller leaf weights force the tree to learn a very generalized representation of the training data.

To transform the problem of optimizing the objective function into a simpler problem of minimizing a quadratic function, Chen & Guestrin (2016) perform multiple steps. They apply the second order Taylor approximation and perform further steps to obtain the following approximate representation of the objective function (4.3):

$$O^{(t)} \approx \sum_{i=1}^{n} \frac{1}{2} h_i \left(e_t\left(x_i\right) - g_i/h_i\right)^2 + \Omega\left(e_t\right) + c, \tag{4.6}$$

where g_i and h_i are the first and second order derivatives of the MSE loss function and c is a constant term. The new representation of the objective function can be used to find the optimal weights w_i that minimize the loss function at each step.

The last remaining problem is to find the optimal cascade of questions to ask for each tree. More precisely, this refers to finding the features and values to define the splits at each node of the tree. Chen & Guestrin (2016) propose two algorithmic solutions to the problem. First, an exact greedy algorithm to solve this problem by enumerating all possible splits and evaluating each option with the help of equation (4.6) and secondly, an approximate algorithm. The approximate algorithm is chosen for this study since it is best suited to handle the amount of training data within the experiments. The basic idea of the approximate algorithm is to find promising possible splits based on the given feature distributions, resulting in improved time efficiency by not having to evaluate all possible splits.

Lastly, the clever system design of Chen & Guestrin (2016) further differentiates between classical gradient boosting and XGBoost. They introduce the concept of blocks in their work. Because data need to be put in order multiple times during the approximate

split finding algorithm, data are stored in blocks, where each column is already sorted by the corresponding feature value. The blocks can be divided and used for parallel computing of the approximate split finding algorithm.

When applying XGBoost to a specific data set, a multitude of hyperparameters can be adjusted to achieve the highest possible accuracy. Throughout the thesis, we will use a Tree-structured Parzen Estimation (TPE) (Bergstra et al. 2013) as a sequential model-based optimization approach. This means that models are constructed sequentially to approximate the performance of the model for a set of hyperparameters based on historical measurements. TPE tries to estimate the underlying relations between a quality measure and the hyperparameters by exposing the underlying expression graph of how a performance metric is influenced by the hyperparameters. For this thesis, the Python implementation of TPE within the Optuna framework is used for the experiments (Akiba et al. 2019) with the exact number of iterations and the selection of the validation set being explained for the individual experiments in Chapter 5.

From Raw Images to Histograms and Feature-Based Representations

The downloaded satellite images form input sequences $\mathbf{d} \in \mathbb{R}^{34\times H\times W\times 11}$, where H and W are the variable height and width of the images, 34 is the number of 8-day episodes and 11 is the number of bands. H and W are variable throughout the data, as the size of the counties varies. Depending on whether an end-of-year prediction with information available covering the whole growing period or an in-year prediction with information available to a certain point is performed, the corresponding time frames of the remote sensing data are selected. The satellite images are cropped to the county borders and masked with the Cropland Data Layer (CDL). Pixels showing non-zero values are distributed unevenly. A common workaround is to process each band of the images into histograms with evenly distributed bins (You et al. 2017). This approach reduces the data size dramatically and still conserves sufficient information for deep learning approaches to learn expressive representations of the data.

However, since the bin values of the histograms are standardized over all images on the one hand and pixel values vary significantly between different images on the other, the resulting histograms show bins filled with zero value entries. The value distributions of the satellite bands mostly follow skewed normal distributions, allowing for an approximation of the value distribution of each band of an image by a normal distribution parameterized by three values, namely the median and the values marking the 20% and 80% quantiles. The choice of features is well suited to describe the skewed normal distributions. Although the median is an outlier resistant approximation of the expected value of the distribution, a small and a high quantile value are useful to monitor the slopes to the left and to the right of the median. The experiments showed that the specific values 20% and 80% are a good choice for the task. This is the feature-based representation that is used as an input for XGBoost. Intuitively, it is possible to model remote sensing data as skewed normal distributions, since soybean acres within proximity to each other will show similar surface reflections and are exposed to similar weather conditions. An example is shown in Figure 4.2. The sparse heatmap depicts the initial data. The orange bars within the histogram representation show how the three selected values describe the underlying skewed normal distribution.

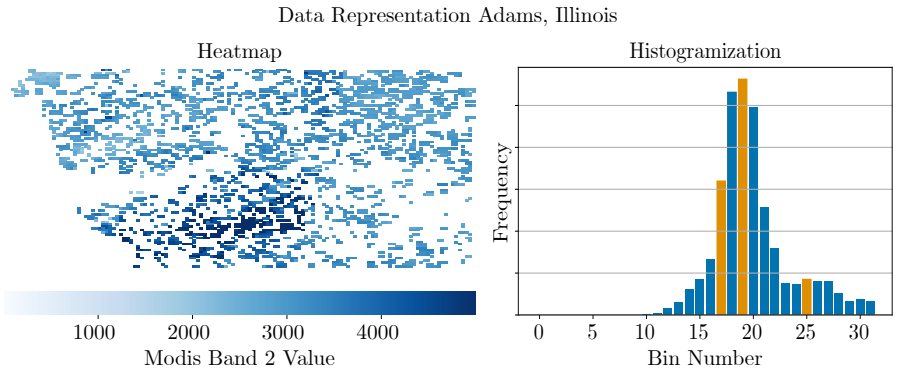


Figure 4.2: Left: Heatmap for band 2 of the masked MODIS surface reflectance data, captured between day 105 and day 113 of 2020 in Adams, Illinois. Right: The histogram showing the relative frequencies of the same data. The bins depicting the 20% quantile, the median and the 80% quantile are highlighted in orange and used as input features for predictions. Figure taken from Huber et al. (2022).

It is important to note that the proposed reduction in dimensionality is based on the assumption of permutation invariance. This means that the prediction of future yields relies more on the value of the non-cropped pixels than on their location. As You et al. (2017) point out, this assumption ignores the likely dependencies of the target output on position-bound features such as soil properties or elevation. You et al. (2017) therefore combined their deep learning approaches with a deep Gaussian process to integrate the missing spatio-temporal information, resulting in only a small improvement of 3.9% in terms of RMSE. Using a Gaussian process for the prediction of yield is more important when fewer data points are available (Kaneko et al. 2019). This must be related to the significant reduction in dimensionality by removing spatial information from the data. Finally, all the values of every band from one sequence are concatenated. Multiplying the 3 values of the feature-based distribution representation of all 11 bands over 34 time steps results in an initial input vector $x \in \mathbb{R}^{1122}$. Since classic machine learning models are able to take advantage of handcrafted features, another seven features are added. Positional awareness is added to the prediction by including the latitude and longitude of the respective centers of the counties. Furthermore, awareness of time is raised by including the year, the sample is taken from, and the number of years passed since 2003. The county-wise average yield computed on the previous years is added to incorporate spatial dependencies. Lastly, the y-intercept and the slope of a linear regression fitted to the historical yields in the data set are included. These features help to explain a growing trend in average soybean yields over the years.

State-of-the-Art Approaches to Soybean Yield Prediction

The XGBoost-based approach is compared with two deep learning approaches to soybean yield prediction. The CNN network presented by You et al. (2017) and the CNN-LSTM hybrid network structure presented by Sun et al. (2019). Both approaches convert the original satellite images into image histograms of each band with evenly distributed bins.

For the histogramization of the bands provided by the MODIS products, data pro-

cessing follows the limits given by You et al. (2017). The original work did not use the Daymet data, but in this study they are included in both approaches for a fairer comparison. Furthermore, the theoretical minimal and maximal values are used as the boundaries for the histograms of the bands provided by the Daymet product. The hyperparameters described in the respective work for both networks are used, including an early stopping criterion when the validation score shows no improvement for 10 epochs. The results are averaged over five runs to account for randomness. The CNN network implementation is taken from a publicly available GitHub repository (Tseng 2022) and the CNN-LSTM hybrid approach from Sun et al. (2019) is implemented from scratch. As for XGBoost, a new model is trained for each year with access to information from all previous years. A random 10% split of the training data is used for validation and all data are normalized by subtracting the mean of the training data.

The CNN structure of You et al. (2017) is very straightforward, consisting of seven convolutional layers of varying sizes with stride-1 or stride-2, followed by a fully connected layer with 2048 neurons for the final prediction. The stride-1 layers serve the purpose of a classical convolution. The stride-2 layers are important to reduce the dimensionality throughout the network, since no pooling layers are used. The same network will be used in Section 4.5, where this will be the basis of our transfer learning efforts.

The CNN-LSTM network of Sun et al. (2019) applies a CNN structure to the input data to learn a meaningful representation that is used as input for a standard LSTM network. The CNN structure consists of two blocks of convolution, batch normalization, and max pooling that are concatenated and used for each of the 34 time steps separately. The output is flattened and serves as an input for the LSTM network with a hidden layer size of 256. Lastly, the output of every iteration is passed through a fully connected layer with 64 neurons and concatenated with each other. The final result is calculated using a dropout layer with a probability of 0.5 before using the last fully connected layer for the singular output.

4.2 Grouped Shapley Values for Explaining Yield Prediction Models

The pipeline for applying XGBoost to yield prediction finally results in a random forest used to make annual yield predictions. In this section, we present the idea of the Grouped Shapley Value (GSV). The contents of this section are previously published (Huber et al. 2023). The described approach is divided into multiple steps (1) definition of GSV for general cooperative games, (2) transfer of GSV to gain local explanations for machine learning models, (3) use of local explanations to gain global understanding, and finally (4) calculation of local explanations in polynomial time for tree structures.

The Value of Predefined Coalitions in a Cooperative Game

The Shapley value is a concept first defined in economic game theory and is defined to solve the fair distribution of resources in a cooperative game. A cooperative game is a tuple (P, v), where $P = \{1, 2, \ldots, p\}$ is the finite set of players and $v : 2^P \to \mathbb{R}$ is the value function. A subset of players P is called a coalition, and the value function assigns

a value to each coalition of players. The Shapley value $\varphi_i(v)$ for a player i and a value function v represents the average contribution of a player to all possible coalitions that players in P can form (Roth 1988) and is defined as:

$$\varphi_i(v) = \sum_{S \subseteq P \setminus \{i\}} \frac{|S|!\,(p - |S| - 1)!}{p!} (v(S \cup \{i\}) - v(S)). \tag{4.7}$$

To better understand the Shapley values, we can look at the fraction at the beginning of Equation (4.7). The fraction evolves from the original Shapley value definition, which is made by permuting the set of players and averaging the difference when evaluating the value function over the coalition with all players that precede a player i in the given order with and without the player i itself. The number of players preceding in our formula coincides with the number of players in the set S, having $|S|!$ possible orders. Similarly, the players who succeed the player i have $(p - |S| - 1)!$ possible orderings. Normalizing with all possible $p!$ permutations results in the factor in Equation (4.7). For our approach, we extend the classic Shapley value formula by allowing players to form predefined coalitions before the game starts. Players in a predefined coalition will be evaluated together and will never be separated. To notate the predefined coalitions of players, we assume a complete partition of the set of players $\mathcal{C} = \{C_1, \ldots, C_k\}$, where each C_i is a nonempty subset of P and represents a different predefined coalition with $\cup\mathcal{C} = P$ and $\forall i, j : C_i \cap C_j = \emptyset$. We note that these restrictions on the predefined coalitions do not allow a direct comparison of different feature groups that share some features, but we will give an outlook on how to solve this problem in the discussion in Section 5.2.

To obtain GSV, the classic definition of Shapley values is restricted to only average the contribution of the group C_i to all possible subsets S that can be built from predefined coalitions within $\mathcal{C}$. Therefore, the GSV $\varphi_{C_i}(v)$ for a predefined coalition C_i depending on the value function v can be defined as:

$$\varphi_{C_i}(v) = \sum_{S \subseteq \mathcal{C} \setminus \{C_i\}} \frac{|S|!\,(k - |S| - 1)!}{k!} (v(\cup S \cup \{C_i\}) - v(\cup S)), \tag{4.8}$$

where $\cup S$ describes the union of all selected sets of $\mathcal{C}$ that are in S. We note that this definition allows the predefined coalition of players to have varying sizes, which is very useful in terms of feature importances, since natural groupings are mostly related to the origin of the feature and vary throughout most data sets. When comparing Equation (4.8) with Equation (4.7), we see that the player i is now replaced by the predefined coalition C_i, as the entire set of players is seen as a singular player. Similarly, the set S that is used to sum all possible coalitions is always a subset of $S \subseteq \mathcal{C} \setminus \{C_i\}$. This means that again, predefined coalitions are always evaluated together, and the players within are never split when calculating the marginal contributions. Since the equation is a direct extension of the classical Shapley value, all desirable game-theoretic properties still hold (efficiency (4.9), symmetry, dummy variable, and additivity). Most importantly, the efficiency axiom ensures that the Shapley value precisely distributes the gain produced

by the coalition consisting of all players among all players.

$$\textbf{efficiency} : \sum_{C_i \in \mathcal{C}} \varphi_{C_i}(v) = v(\mathcal{C}). \tag{4.9}$$

Later, the axiom of efficiency translates into local explanations that always add up to explain the output value of the model for the explained data point. Exactly this important axiom of efficiency is not preserved in game-theoretic approaches to extend Shapley values on predefined coalitions that are proposed in the literature (Grabisch & Roubens 1999, Marichal et al. 2007, Flores et al. 2019). Therefore, these game-theoretic approaches are not applicable towards explainable machine learning, as the explanations would not add up to explain the model outcome. The main difference between the GSV approach presented in this work and the related work mentioned is the treatment of players that are not in the predefined coalition that is subject to the Shapley value calculation. While in our approach and Equation (4.8) we expect the other players to play the game restricted to their respective groups, the other approaches focus on a comparison where the other players are still seen as individuals. This is helpful when the subject of the study is to find which players gain value from joining a predefined coalition. Our approach instead excels in comparing the performance of disjoint groups, as they appear in our yield prediction scenarios, where the predefined coalitions are given by the structure of the input data. A naive solution to aggregate Shapley values within a group of players is to add individual Shapley values as we did in an early iteration of explaining our yield predictions (Huber et al. 2022). To show that this option is not appropriate and produces counterintuitive results in a game-theoretic content, we analyze a minimal example based on the classic illustrative glove game (Aumann & Shapley 1974). Later in this thesis, we give an example for the calculation of grouped Shapley values, where we expose the differences in both approaches to explain a decision tree.

The glove game example: Within the glove game, three players $P = \{1, 2, 3\}$ trying to complete a pair of gloves. The player 1 and the player 2 each have a left glove, while the player 3 has a right glove. The value function $v(S)$ is evaluated with the value 1, if the set S contains a matching pair of gloves and with the value 0 otherwise. Calculating the classic Shapley value, we obtain $\varphi_1(v) = \varphi_2(v) = \frac{1}{6}$ and $\varphi_3(v) = \frac{4}{6}$. The results follow our direct intuition that the player 3 is the most important player in the game, since it is the only player who can complete a pair of gloves. Our observation changes when players 1 and 2 form a predefined coalition. This means $C_1 = \{1, 2\}$ and $C_2 = \{3\}$. Both groups should be valued equally within the game, as having multiple left gloves does not increase the value function, and only a combination of C_1 and C_2 can build a pair of gloves. Using equation (4.8) we observe that our definition of grouped Shapley values follows this intuition by valuing $\varphi_{C_1}(v) = \varphi_{C_1}(v) = \frac{1}{2}$, while when summing the initial values $\varphi_1(v) + \varphi_2(v) = \frac{2}{6}$ we would undervalue the coalition of players 1 and 2. Note that we can increase the gap between both approaches by adding and grouping more players who own a left-hand glove to the game.

From Grouped Shapley Values to Local Explanations.

To use GSV to better understand yield prediction, we define a game (P, v) in the context of a machine learning model M trained on a data set $X \in \mathbb{R}^{n,m}$ with targets $y \in \mathbb{R}^m$. That is, the data set consists of n features $\{f_1, \ldots, f_n\}$ and m data points used to create the model. We obtain local explanations for a fixed data point $x \in X$ by interpreting each of the features as a player in a cooperative game, so $P = \{f_1, \ldots f_n\}$. For a subset of features $S \subseteq P$, we want the value function $v(S)$ to represent the answer of the model M to the data point x, assuming only the feature values of the data point from features in S are known. For any predefined coalition $C_i \subseteq P$ and set $S \subseteq C \backslash \{C_i\}$, the difference between $v(\cup S)$ and $v(\cup S \cup \{C_i\})$ describes the change in the output of the model, assuming additional knowledge of the values of the features in C_i. The idea of the GSV is to average contributions over all possible combinations of other predefined coalitions S and to give an estimate of the impact of specific values of the features in C_i to the model output.

To give an estimate of the model answer based on limited access to features, we will calculate the expected answer of the model M for a data point x, where only the features in S are known: $\mathbb{E}[M(x)|S]$. This is known as a conditional value function, as we estimate the model output under the condition that only certain features are known. For this discussion, we will focus on a value function that can be defined for decision trees. This will be useful for yield prediction, since we have just established a pipeline to use XGBoost for yield prediction that sequentially results in a random forest to make the predictions. To estimate the model's answer of a singular tree given a subset of features, we will traverse the tree. If, while traversing, we find a feature F, which is not included in S, we estimate the average model answer from data points in our training data set that are similar to x in relation to the set S. Similar data points are defined as points that induce the traversing of the tree equal to the data point x, for every feature in S. For the feature F that is not in S we calculate the weighted average of the model answer, according to the number of similar data points that follow the two possible branches of the tree. The procedure was first described by Lundberg et al. (2020) and can be calculated using the recursive Algorithm 1. For the algorithm, we assume the model M to be a tree that can be displayed as multiple lists. Including *val* (values) to store the output value of the node, if the node is a leaf, l (left) and r (right) to store the index of the left and right child nodes, f (features) and t (thresholds) to indicate which feature to split at what threshold for the according node, and finally c (cover) to store the number of training data points that follow the node. The root is always saved at the index 0.

We give exemplary calculations in Figure 4.3 and put emphasis on handling the unknown feature "Rain" in part a) and "Temp_day" in part b). With Algorithm 1, we can calculate the value function of Equation (4.8) for any given data point and subset. By iterating over all the necessary subsets of features and building the sum, we can calculate the GSV. Later in this section, we will describe a further procedure for calculating Equation (4.8) in polynomial time.

Algorithm 1: Conditional value function $v_{M,x}(S)$. First introduced by Lundberg et al. (2020)

Input : S, a subset of features
x, a data point
$M = \{val, l, r, f, t, c\}$, the model consisting of values, left children, right children, split feature, threshold and cover
Output: $v_{M,x}(S) = E\left[M(\boldsymbol{X}) \mid \boldsymbol{X}_S = x_S\right]$, the expected model output for the data point x given only access to the feature values in S

```
Function traverse(node):
    if node is internal then
        if f[node] ∈ S then
            nextnode = next node according to x
            return traverse(nextnode)
        else
            leftpath = traverse(l[node]) * c[l[node]]
            rightpath = traverse(r[node]) * c[r[node]]
            return (leftpath+rightpath)\ c[node]
        end
    else
        return val[node];
    end
EndFunction
return traverse(0)
```

From Local Explanations to Global Understanding.

The approach described for GSV is capable of obtaining local explanations for any yield prediction model, that is, predictions that can explain the impact of each feature for a specific data point. In the context of yield prediction, a data point will be a vector containing the values of the input features that are used to predict the yield for one instance in one year. To continue, we want to have access to a global understanding of the model. Only then can we decide to drop features of low importance or analyze general patterns within the data that will help us to understand the yields. The low dimension of the GSV allows a clear representation of the importance of the features. The first step is to calculate the local GSV for a variety of data points. We can then utilize specialized swarm plots for a joint representation to get an idea of not only the magnitude of impact each feature group possesses, but also the impact of high and low feature values on the final prediction. Swarmplots are plots where each point is placed according to its value on the y-axis. If multiple points have similar values, an area gets crowded and the points are spaced out, creating visual clusters. The plots are well suited for our task, because they excel in visualizing the distribution of data points in a small to medium amount of categories, like we have reached by grouping our features. Together with applying a hue to the points, we can reveal meaningful patterns in the data. The information gained can then serve as baseline for further experiments to decide whether the machine learning task at hand is reliant on a specific group of features and can lead to decisions like, e.g., not buying a certain sensor or not using human resources to capture in-field information that often. We build the swarm plots as follows: After grouping the features, we decide on an aggregated value to represent the magnitude of the features in the group. For similar features, such as features captured from the same sensor in multiple timeframes, the mean value of all features serves this purpose.

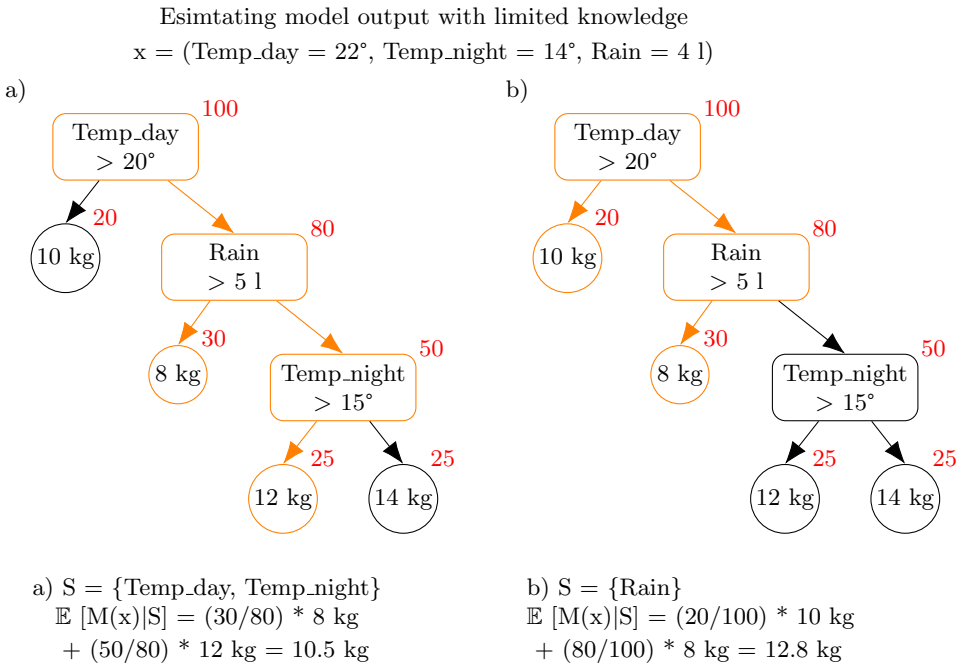


Figure 4.3: Two calculations of expected the model output $\mathbb{E}[M(x)|S]$ for a fixed data point x, feature-subset S and model M as it is used for Shapley values calculation. The number of data points covered by each node is indicated by the number next to it, and fulfilling the condition of a node results in following the right path of the tree. The same path ending in the prediction labeled "8 kg" is used to estimate the model answer of two different feature-subsets S. Figure taken and altered from (Huber et al. 2023).

Since the number of groups is limited, we can even visualize multiple swarm plots in one figure, where each swarm represents a feature group. The x-axis is then used to show the GSV of the regarding group, indicating this feature group's impact on the prediction in the grand scheme. Finally, after normalizing the representative values for all groups, we can use the hue of the individual points to highlight how the high and low values of the feature groups influence the prediction of the model. Examples are shown later in Figures 5.1 and 5.2. Having the GSV calculated can help us understand the impact of the group of features C_i on our model for the specific data point x. Each GSV can be interpreted as the difference made for our model that the features in C_i are valued within x the way they are, in comparison to what the model would output, if these values were unknown. The higher the absolute value, the more impact is attributed to the feature group.

Grouped Shapley Values on Decision Trees (conceptual)

In general, the question of calculating the Shapley values is known to be NP-hard (Conitzer & Sandholm 2004). The usage of predefined a priori coalitions is capable of reducing the complexity of the task by reducing the number of summands in Equation (4.8) compared to the classic Shapley value. We have already established how to estimate the value function $v(S)$ for a model M and a fixed data point x and how to use

it to calculate Equation (4.8). Understanding the naive algorithm alters the way of formulating an algorithm capable of solving Equation (4.8) in polynomial time. Lundberg et al. (2020) give a polynomial time algorithm for the calculation of the classic Shapley value feature importances on tree structures. On the basis of their work, we are able to formulate an algorithm to calculate the GSV feature importances in polynomial time. To better understand the algorithm, we start with a conceptual representation of the idea with Algorithm 2, where we abandon the exact weight update. The next subsection will give an example calculation and finally the exact version of the algorithm can be found later in this section in Algorithm 3.

Algorithm 2: Polynomial Group Shapley Values for Trees (Conceptual)

```
   Input  : x, a data point
            M, the tree model
            C, the partition of features that form the predefined coalitions
   Output: φ, the grouped Shapley values
 1 Function GroupTreeShapleyValue(Datapoint: x, Model: M, Coalitions: C):
 2     φ = array of len(C) zeros  // Storage for GSV
 3     Function Expand(currentNode, path, weigths):
 4         path, weights = Update_weights(currentNode, path, weights)
 5         C_i = Group of currentNode.getFeature()
            // Get the group of the node's split-feature
 6         if currentNode is an inner node then
 7             if a previous Feature F along the path is also in C_i then
 8                 UNDO the Update_weights for F  // Subset sizes remain the same
 9             Child1 = Traverse further following the values of x
10             Child2 = Traverse further the other child
11             Expand(Child1, path, weights)
12             Expand(Child2, path, weights)
13         if currentNode is a leaf then
14             for node in path do
15                 C_j = Group of node.getFeature()
16                 w_frac = weights according to subsets along the path in regard to C_j
17                 w_pos = weights of splits according to features in C_j
18                 w_neg = weights of splits without information of features in C_j
19                 φ [j] += w_frac * (w_pos − w_neg)* currentNode.getValue()
20             end
21     EndFunction
22     EXPAND(root, path = [], weights = [])
        // Start at root node with empty path and no weights
23     return φ
24 EndFunction
```

We have already described an intuition for calculating Equation (4.8) by iterating over all possible subsets of the sum and estimating the answer of the value function. For each subset S we have to traverse multiple paths of the tree, as highlighted in Figure 4.3, since every time we need to split at a feature not in S, we continue to build the weighted average over the two possible following children. Similarly, we use each path of the model M in multiple calculations for the sets of Equation (4.8). In the examples in Figure 4.3, we see the path ending in "8 kg" traversed for two different subsets S. The main idea of the polynomial algorithm is to traverse the entire tree only once while keeping track of the contribution that each individual path has to all possible GSVs. At the end of the traversal of each path, the grouped Shapley values are updated accordingly. The weights are determined by three different factors that need to be tracked and updated throughout the algorithm. First, the sizes of the possible sets S as necessary for the

factor in Equation (4.8) (noted as w_frac in Algorithm 2). Second, the fraction of training examples that follow the branches, as explained in the example in Figure 4.3. Third, for every group C_i we want to calculate the GSV for, we need a sign depending on whether the path traversal assumes knowledge of the features in C_i (positive) or not (negative). The weights are updated consecutively while traversing the tree. We focus on line 7 of the algorithm, where we need to check if the feature group was already represented during the path we traverse currently. If this is the case, the structure of subsets leading down to the path does not change, since all features within a group are treated as a unit.

The algorithm was first defined in (Lundberg et al. 2020) for classic Shapley values. We altered the algorithm to calculate the grouped Shapley values. The correctness of our algorithm follows directly from the work of Lundberg et al. (2020), since we only consider multiple features to be handled as if they were the same player within the equation. Since the check of the occurring groups can be made via a lookup table, the run time is still bounded by $O(TLD^2)$, with T being the number of trees within the random forest. This is a factor since we need to execute the algorithm for every individual tree. L refers to the maximum number of leaves within the trees and D is the maximum depth of the trees. To understand the runtime, we have to look at the exact version of the algorithm, Algorithm 3. Calling the *Expand* function in line 3 calls the *Unwind* function in a loop that is bound by the length of the regarding path, which is bound by the maximum depth D. *Unwind* itself is also bound by the depth D, explaining the factor D^2 of complexity. All other loops are bound by the length of the path m and therefore by D. The *Expand* call with runtime D^2 has to be done for every path, which is bound by the number of leaves L, causing the runtime $O(LD^2)$ for one tree and $O(TLD^2)$ for all trees of the forest.

Grouped Shapley Value Algorithm Example

In this section, we analyze the model shown in Figure 4.4, exposing differences in the Shapley value feature attribution for grouped Shapley values compared to simply summing up the individual values. We will compare the case where each feature is seen as an individual with the a priori defined groups $\mathcal{C} = \{C_1 = \{F1, F3\}, C_2 = \{F2\}\}$. In both cases, we will use the algorithm for fast calculation of the (grouped) Shapley values to calculate the Shapley value of the feature $F2$ and then the group C_2 that contains the feature $F2$. To improve readability and since x, v and M are fixed throughout the example, we will write φ_{F2} instead of $\varphi_{F2}(v)$. As the sum of all Shapley value attributions will be the same in both scenarios, since we are still explaining the same model by the axiom of efficiency, when $\varphi_{F2} \neq \varphi_{C_2}$, then $\varphi_{F1} + \varphi_{F3} \neq \varphi_{C_1}$, where C_1 contains both $F1$ and $F3$. We can use Algorithm 2 to calculate φ_{F2}, as the ungrouped Shapley value is just a special case of the GSV, where every group contains a single player. To calculate φ_{F2}, following Algorithm 2 we will calculate the positive and negative contribution of each path in the model M towards the final Shapley value. Each path is represented by the leaf node in which it ends. A path can only make a positive contribution to a Shapley value φ_{Fi}, when for each appearance of the feature Fi as a split feature, the path traverses the tree according to the data point x. This is because in the Shapley value calculation (Equation (4.8)), the sign of the value function is only positive for

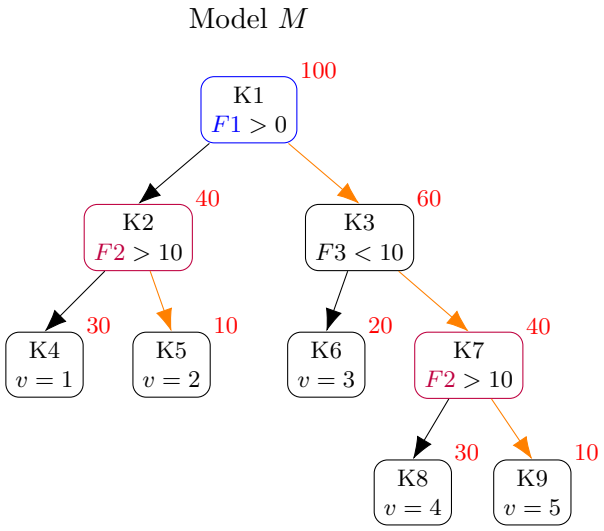


Figure 4.4: An example model for a prediction problem consisting of three features $F = \{F1, F2, F3\}$. The first row in each node shows its name, the second shows the split condition for internal nodes and the output value for leaf nodes. If the condition of an internal node is met, we follow the right path of the tree. The example data point $x = [5, 15, 5]$ follows the right path in each node, as is indicated by the orange arrows. Lastly, the red number shows the number of data points covered by each node.

sets that include the according feature Fi. For the negative contributions, we track the weights according to the cover for each path to account for two different factors. First, we trace the fraction at the beginning of Equation (4.8) according to the number of feature groups included in the path. This value is presented by w_frac in line 16 of the algorithm. Second, we track the fraction of training examples following the path to account for coalitions where we assume some features along the path are unknown, represented by w_pos and w_neg in lines 17 and 18 of the algorithm.

Starting with the example, for the path ending in K4, this means that we have no positive contribution, since we do not reach K4 when $F2$ is known. Regarding the weights, we obtain $\frac{1}{2}$ as the factor in front of the Shapley value equation, since two different features are present in this path. The two possibly negative contributing coalitions to the path are then the one where $F1$ and $F2$ are unknown, represented by the $\frac{3}{10}$ of training examples following this path according to the cover, and the coalition where $F1$ is known and $F2$ is unknown, giving no contribution as in this case we cannot reach K4 due to the split in K1. For the path ending in K5, we have a positive and a negative contribution, since we can reach K5 when we assume knowledge about the feature $F2$. The calculations follow the same logic. The exact calculation of the contributions is as follows:

$$\begin{aligned} \text{K4:}\quad \text{pos} &= 0 \\ \text{neg} &= \tfrac{1}{2} * \tfrac{3}{10} \\ \varphi_{F2} &= 0 - \tfrac{3}{20} * 1 = -\tfrac{3}{20} \end{aligned} \qquad \begin{aligned} \text{K5:}\quad \text{pos} &= \tfrac{1}{2} * \tfrac{4}{10} \\ \text{neg} &= \tfrac{1}{2} * \tfrac{1}{10} \\ \varphi_{F2} &= -\tfrac{3}{20} + (\tfrac{4}{20} - \tfrac{1}{20}) * 2 = \tfrac{3}{20} \end{aligned}$$

Resuming the algorithm with the calculation of the contributions of the path ending in K6, we see that on line 19 of the Algorithm 2 the Shapley value is only updated when the regarding feature was used as a split feature along the path. As this is not the case for the path ending in K6 and feature $F2$, we do not have updates. For the path ending in K8 and K9 we now find the fractions $\frac{1}{6}$ and $\frac{2}{6}$ to represent the fraction in the Shapley value formula, as here three features are involved in the calculation. The weights representing the amount of training data that follows the path when features are assumed to be unknown are calculated as above. This leaves us with:

$$\begin{array}{llcl} \text{K6:} & \text{pos} & = & 0 \\ & \text{neg} & = & 0 \\ & \varphi_{F2} & = & \frac{3}{20} + 0 * 3 = \frac{3}{20} \end{array} \qquad \begin{array}{llcl} \text{K8:} & \text{pos} & = & 0 \\ & \text{neg} & = & \frac{2}{6} * \frac{3}{10} + \frac{1}{6} * \frac{3}{6} + \frac{1}{6} * \frac{6}{10} * \frac{3}{4} + \frac{2}{6} * \frac{3}{4} \\ & \varphi_{F2} & = & \frac{3}{20} - \frac{61}{120} * 4 = -\frac{113}{60} \end{array}$$

And finally, to represent the complete tree and get the Shapley value:

$$\begin{array}{llcl} \text{K9:} & \text{pos} & = & \frac{2}{6} * \frac{4}{10} + \frac{1}{6} * \frac{4}{6} + \frac{1}{6} * \frac{6}{10} * \frac{3}{4} + \frac{2}{6} * 1 \\ & \text{neg} & = & \frac{2}{6} * \frac{1}{10} + \frac{1}{6} * \frac{1}{6} + \frac{1}{6} * \frac{6}{10} * \frac{1}{4} + \frac{2}{6} * \frac{1}{4} \\ & \varphi_{F2} & = & -\frac{113}{60} + (\frac{61}{90} - \frac{61}{360}) * 5 = \frac{79}{120} \approx 0.658 \end{array}$$

Now we calculate the same example but with pre-defined groups of features. The groups are defined as follows: $\mathcal{C} = \{C_1 = \{F1, F3\}, C_2 = \{F2\}\}$. First we note, that there is no change in the calculations for K4, K5 as they represent paths not including $F3$. For the path ending in K6, the contributions are still 0, as $F2$ is not present. However, for K8 and K9 we get different results. On the one hand, we have only two feature groups present in the path, which means that w_frac is now $\frac{1}{2}$ and we no longer have a coalition where $F1$ is present but $F3$ is not, which makes a difference in the calculation. In a formal representation, we get the following:

$$\begin{array}{llcl} \text{K8:} & \text{pos} & = & 0 \\ & \text{neg} & = & \frac{1}{2} * \frac{3}{10} + \frac{1}{2} * \frac{3}{4} \\ & \varphi_{C_2} & = & \frac{3}{20} - \frac{21}{40} * 4 \end{array} \qquad \begin{array}{llcl} \text{K9:} & \text{pos} & = & \frac{1}{2} * \frac{4}{10} + \frac{1}{2} * 1 \\ & \text{neg} & = & \frac{1}{2} * \frac{1}{10} + \frac{1}{2} * \frac{1}{4} \\ & \varphi_{C_2} & = & -\frac{39}{20} + (\frac{7}{10} - \frac{7}{40}) * 5 = \frac{27}{40} = 0.675 \end{array}$$

This example again proves the difference between GSV and simply building the sum of individual Shapley values. And while the difference in our example is relatively small with $0.675 - 0.658 = 0.017$, we can use the idea of the simple example to create arbitrarily large differences by increasing the value of the leaf node K9. Furthermore, this example can help us to understand how the fast calculation for grouped Shapley values utilizes the given tree structure. We can omit any operations that scale with the powerset of the players and substitute it by an algorithm that loops over the different paths within a decision tree.

Grouped Shapley Values on Decision Trees (detailed)

Based on the work of Lundberg et al. (2020) we present Algorithm 3 as a precise description of Algorithm 2. Regarding the notion within the algorithm, as above, we assume that the model M is a tree that can be displayed as multiple lists. Including *val* (values)

to store the output value of the node, if the node is a leaf, l (left) and r (right) to store the index of the left and right child nodes, f (features) and t (thresholds) to indicate which feature to split at what threshold for the according node, and finally c (cover) to store the number of training data points that follow the node. The root is always saved at the index 0. The additional variable m is used to store the path of the unique feature groups along the path, represented by one feature per group. Together with the path m, we store four attributes. (1) The feature index f, (2) the fraction of paths, where this group is not in the set S that flow through the branch z, (3) the fraction of paths, where this feature group is in the set S - o, and finally (4) the weight w, which keeps track of the weights in front of Equation (4.8). Within the algorithm, we access arrays through dot notation and $m.d$ represents the whole vector of features that are traversed so far. Lastly, the values p_z and p_o track the fraction of added contributions, depending on the current feature as represented in the subsets.

4.3 Discussing Shapley Values for Feature Selection

In the previous section, we established how Shapley values can be a valuable tool for explaining yield prediction models, especially under consideration of naturally available feature groupings. With this in mind, a natural next question arises: Can we use Shapley values as a feature selection tool? More precisely, can we reduce the dimension of the input feature space, while preserving the accuracy of our given model. In this section, we lay the theoretical foundation for an application-driven analysis later in this thesis. We first define 4 necessary conditions for defining a Shapley value application in machine learning to circumvent known problems for Shapley value feature selection. After this we unpack the model averaging problem, where, by definition of the Shapley value, features that should not be selected during a feature selection procedure can alter the final selection. We give an algorithm to obtain a feature selection that follows the same core idea as Shapley values but has no problems with model averaging, called Conditional Feature Selection (CFS). For simplicity, we focus our discussion mostly on normal Shapley values as a special case of GSV, but the results can be directly related. Parts of this section are already published in our previous research (Huber & Steinhage 2024d).

Necessary Conditions for Defining Shapley Values for Feature Selection

In the previous section, we defined GSV and applied the game-theoretic principle of Shapley values to machine learning. However, within this definition, it is necessary to choose a value function that gives each subset of players a value. Throughout this thesis, so far, we have chosen a value function as defined by Algorithm 1, but there are other possible choices to consider, such as the selected value function being integral to a successful use of the Shapley values for feature selection. We define conditions to define a value function that enables Shapley value feature attributions to be well suited for feature selection. The conditions are based on recent criticism of the Shapley values (Kumar et al. 2020, Huang & Marques-Silva 2023, Sundararajan & Najmi 2020, Fryer et al. 2021) and avoid the problems that the respective works have uncovered. Although the four mentioned works regarding this topic focus on defining a value function for

Algorithm 3: Polynomial Group Shapley Values for Trees (Detailed)

Input : x, a data point,
$M = \{\text{val, l, r, f, t, c }\}$, the model consisting of values, left children, right children, split feature, threshold, and cover
$\mathcal{C}$, the partition of features that form the predefined coalitions
Output: φ, the grouped Shapley values

Function *GroupTreeShapleyValue(Datapoint: x, Model: M, Coalitions: $\mathcal{C}$)*:
  φ = array of len($\mathcal{C}$) zeros `// Storage for GSV`
  Function *Expand*(j, m, p_z, p_o, p_i):
    m = Weight_update(m, p_z, p_o, p_i); `// Update m and all fractions to incorporate the growing amount of features included in the path`
    if val_j *is an internal node* **then**
      $hot, cold = (l_j, c_j)$ if $x_{f_j} \leq t_j$ else (c_j, l_j) `// check path according to x`
      $i_z = i_o = 1$
      k = FINDFIRSTGROUP $(m.f, f_j)$ `// Check for group of` f_j
      if $k \neq$ *nothing* **then**
        $i_z, i_o = (m_k.z, m_k.o)$ `// Undo split if the group is already represented`
        m = Unwind(m, k)
      Expand $(hot, m, i_z c_{hot}/c_j, i_o, f_j)$ `// Recursive call for both children`
      Expand $(cold, m, i_z c_{cold}/c_j, 0, f_j)$
    if val_j *is a leaf* **then**
      for $i \leftarrow$ *0 to len(m)* **do**
        $w = sum(\text{Unwind}((m, i).w)$ `// traverse path backwards`
        $C =$ group of f_j
        $\varphi_C = \varphi_C + w\,(m_i.o - m_i.z)\,val_j$ `// Update Shapley value`
      end
  EndFunction
  Function *Weight_update*(m, p_z, p_o, p_i):
    $l, m = len(m), copy(m)$
    subsetsize = 1 if $l = 0$, else subsetsize = 0 `// Check if this is the first call`
    $m_{l+1}.(f, z, o, w) = (p_i, p_z, p_o, subsetsize)$
    for $i \leftarrow l$ *to* 1 **do**
      $m_{i+1}.w = m_{i+1}.w + p_o \cdot m_i.w \cdot (z/l)$ `// Fraction for bigger subsets`
      $m_i.w = p_z \cdot m_i.w \cdot (l - i)/l$ `// Fraction for same size subsets`
    end
    return m
  EndFunction
  Function *Unwind*(m, i):
    $l, n, m = len(m), m_l.w, copy(m_{1\ldots l-1})$
    for $i \leftarrow l - 1$ *to* 1 **do**
      if $m_i.o \neq 0$ **then**
        $t = m_j.w$ `// Run the path backwards`
        $m_j.w = n \cdot l/\,(j \cdot m_i \cdot o)$
        $n = t - m_j.w \cdot m_i.z \cdot (l - j)/l$ `// Undo the calculations within` Weight_update
      else
        $m_j.w = (m_j.w \cdot l)\,/\,(m_i.z(l - j))$
      end
    end
    for $j \leftarrow i$ *to* $l - 1$ **do**
      $m_j.(f, z, o) = m_{j+1}.(f, z, o)$
    end
    return m
  EndFunction
  EXPAND$(root, path = [], weights = [])$ `// Start at root node with empty path and weights`
  return φ
EndFunction

Shapley values that will cause negative examples when applied as a tool for feature selection, we take learning from their counterexamples and extract necessary conditions for defining the Shapley value in a way that they can be used for feature selection. A detailed description of the problems and solutions will be given in the following. The presented conditions will result in a value function well suited for feature selection and explainability on regression trees. The conditions are as follows:

C1: The value function must not be interventional. In general, two families of value functions are considered in the literature, conditional value functions and interventional value functions. Conditional value functions define the expected conditional output of the model at a data point, assuming that only the features of S are known to the model:

$$v_{M,x}(S) = E\left[M(\boldsymbol{X}) \mid \boldsymbol{X}_S = x_S\right]. \tag{4.10}$$

Interventional value functions are motivated by causal inference. They simulate an intervention on the features not in S, denoted as $\bar{S}$, by modeling a distribution $\mathcal{D}$ from the product of the marginal distributions of the features in $\bar{S}$:

$$v_{M,x}(S) = E_{\mathcal{D}}\left[M\left(x_S, \boldsymbol{X}_{\bar{S}}\right)\right]. \tag{4.11}$$

We note that this definition of a value function allows the simulated features of $\bar{S}$ to break with the correlations with the features of S. The negative effects of the interventional value functions for the Shapley values in explainability and feature selection are examined by Kumar et al. (2020). An interventional value function will break down with the correlations within the data set.

For a simple example, we can look at a problem where the feature space consists of two temperature measures of the same day, and we try to predict the temperature of the following day. Naturally, in this scenario, the two features are somehow correlated, as the temperature can only vary so much during one day. The question of choosing a value function now becomes the question of giving estimates of the model behavior when one of the measurements is missing. When choosing a conditional value function, we need to estimate the model output directly without missing information. For an interventional value function, we do not estimate the model output, but estimate the feature value and then use the new fully reconstructed data point to obtain a model prediction. From an application-driven point of view of feature selection, interventional value functions are problematic because of the difficulties of modeling distributions, as it has to be done to estimate the missing feature value from the remaining features of a subset. In our example, when one of the measurements is unknown, for an interventional value function, we need to simulate its value. That could, for example, result in merging measurements that are taken from different seasons and show a wider difference than naturally possible in our data set, breaking the natural correlations of the data. Finally, this leads us to evaluating the model at a data point that is not in the training distribution, where, especially for random forests, the model behavior is unpredictable.

C2: The value function should correctly attribute relevant and irrelevant features. Huang & Marques-Silva (2023) create a value function that allows a Shapley value feature attribution to attribute no value to features that are important for model prediction. In the same scenario, features that are not important to the model output

receive a non-zero attribution value. The choice of a value function to utilize Shapley values for feature selection should avoid these problems.

C3: The value function should consider multiple different training data points when estimating the expected conditional answer of the model. This property is important, since it prevents the evaluation of the value function $v_{M,x}(S)$ to only consider data points where the values of the features in the set S match the values of x. As Sundararajan & Najmi (2020) point out, this can cause problems when the data point x is unique within the data set. This leads to the value function always estimating the model output based on the output of a singular data point and, subsequently, $v_{M,x}(S) = v_{M,x}(T)$ for all $S, T \subseteq P$, making it impossible to distinguish important features.

C4: The value function should allow the calculation of Shapley value feature attributions in polynomial time. The problem of Shapley value calculation is known to be NP-hard. As the number of features can be high, especially when we consider the task of feature selection, the calculation should be feasible in a moderate time frame. Although there exist approximation techniques for the Shapley value calculation (Castro et al. 2009, 2017) that can improve the computational complexity, they add a layer of uncertainty that needs to be considered.

We note that our chosen conditional value function has all the properties explained above, either for groups of features or for singular features. The function is conditional and not interventional (C1). When considering features relevant to the model, we can define a relevant feature as a feature capable of altering the model output. When a feature f is capable of altering the model output, there exists at least one node within the tree that splits on the value of f. So, our value function will give a different result, when evaluated on the sets $S \subseteq P \backslash \{f\}$ and $S \cup \{f\}$, giving a non-zero marginal contribution and finally a non-zero attribution of the feature. Similarly, features that are not relevant and are not considered as split feature in any node within the tree will never get a non-zero marginal contribution and will receive no feature attribution (C2). Furthermore, the chosen value function will consider multiple training instances when estimating the conditional expected answer of the model, as long as at least one relevant feature f is missing from the set S. If a relevant feature is missing, at least two branches of the tree are followed during estimation of the expected answer of the model M. Both branches represent at least one data point in the training data following the path, as otherwise no split at the feature f would have been possible (C3). And finally, the calculation of the Shapley value feature importance is possible in polynomial time (C4), as stated by Lundberg et al. (2020) for the classical definition of Shapley values and for groups of features, as is described in this thesis.

On the Problem of Model Averaging

In the previous subsection, we explained how Shapley values must be utilized for successful feature selection. In this subsection, we discuss the weakness of Shapley values for feature selection, which cannot be fixed by the choice of the value function. To evaluate the impact of model averaging, we give an algorithm that uses the conditional value function defined above to perform feature selection without suffering from model averaging. Even with a carefully selected value function, a feature selection based on

Shapley values cannot be optimal. The nature of Shapley values includes measuring the impact of a feature on every possible subset of features, as can be seen in Equation (4.7). But what is model averaging? In feature selection for prediction models, it is uncertain which features should be considered for the final feature selection. Model averaging is the process of quantifying the influence of all features with respect to predictions (Madigan & Raftery 1994, Raftery 1995). Shapley value feature selections are a model averaging procedure by definition, as Shapley value feature attributions are calculated by considering all possible subsets of features during the calculation of the Shapley value of every singular feature. This results in features that are not in the optimal feature selection can alter the feature selection process. For example, we can assume a scenario with two features $f1$ and $f2$ that explain the data well, when selected together, but poorly when selected alone. When calculating the feature attribution for the feature $f1$, the importance will not be correctly reflected, since for every set S within Equation (4.7) that does not include the feature $f2$, the marginal contribution when adding $f1$ will be low. This results in a Shapley value feature selection that may not select the features $f1$ and $f2$ in the correct position. A real-world example of this behavior is explained in Section 5.3.

Conditional Feature Selection

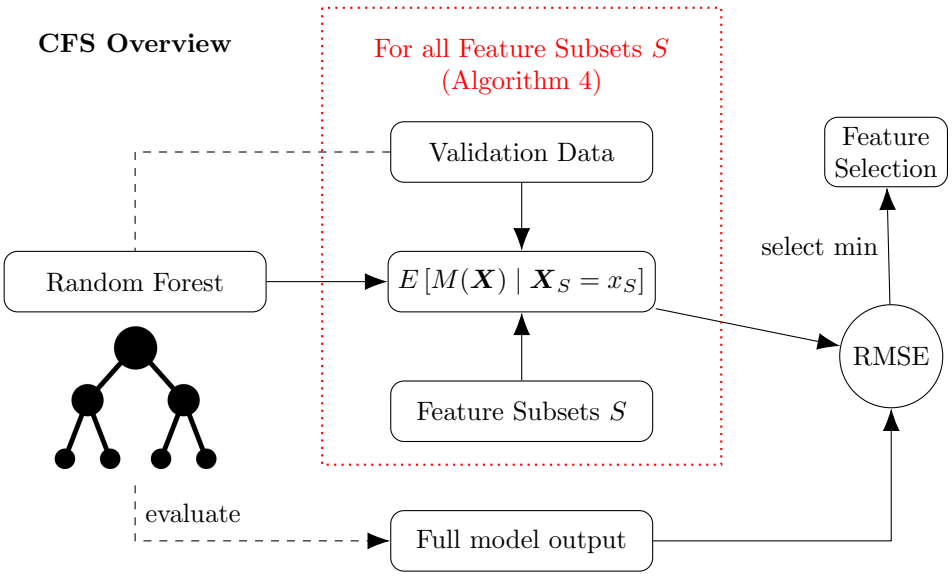


Figure 4.5: Overview of the Conditional Feature Selection (CFS) process to evaluate the problem of model averaging for Shapley value feature selection. Algorithm 4 is used to evaluate the conditional model output for all possible subsets of features on the validation data. A comparison by RMSE with the full output of the full model is used to find the optimal feature selection. This figure was previously published (Huber & Steinhage 2024d)

As a solution to the above problem, we present an algorithm called Conditional Feature Selection (CFS) that allows us to evaluate a feature selection S without considering the features not in S, simply traversing the decision tree once. A visual summary of the procedure is shown in Figure 4.5. Since we will update the estimated model answer for

All Feature Subsets	f_1	f_2	f_3	f_1f_2	f_1f_3	f_2f_3	$f_1f_2f_3$	$\emptyset$
$setswithf_1$	1	0	0	1	1	0	1	0
$setswithoutf_1$	0	1	1	0	0	1	0	1

Figure 4.6: Explanation of the vectors used for the calculation of CFS for an example with three features. Each subset is associated with a fixed position within a vector to store the weights and the output value in Algorithm 4. The vectors $setswithf$ and $setswithoutf$ are calculated for each feature and used to update the weights and output values. Figure previously published (Huber & Steinhage 2024d).

all possible subsets simultaneously, we start by allocating memory for the output of all possible 2^n sets S by creating a vector $out \in \mathbb{R}^{2^n}$. Each set S corresponds to exactly one entry in out, and the vector will be used to synchronize the results of multiple recursive calls of the $traverse$ method, each ending in a different leaf. A possible correspondence of the feature subsets with an order in the vector can be found in Figure 4.6. Furthermore, we use a vector $weights \in \mathbb{R}^{2^n}$ with the same correspondence to the sets S to store weights that are unique for every path of the tree. The weights determine to what extent the value of the path leaf contributes to $v_{M,x}(S)$. Following a path and splitting at a node n with a feature f, for a set S, we observe three cases to update the weight. First, when $f \in S$ and we follow the path according to the value of f in x, the weights remain the same. Second, when $f \in S$ and we follow the path not according to the value of f in x, the weights are multiplied by 0. Third, when $f \notin S$ we need to estimate the behavior of the model based on the fraction of training data points that follow both branches of the path, determined by the cover of the node n and both children. For the following algorithm, we assume pre-processing to generate lists $setswithf$ and $setswithoutf$ for each feature f. $setswithf$ which has the value 1, when $f \in S$ for the set S that corresponds to the same spot in out and the value 0 otherwise. $setswithoutf$ has the value 1, when $setswithf$ has the value 0 at the same spot and the value 0 otherwise. An example of the two vectors is shown in Figure 4.6. When working with vectors, $*$ is the scalar multiplication and $\odot$ is the elementwise multiplication.

We can now use this algorithm to find the best selection of features S for a given criterion. This can be a maximum number of features k to be selected or a maximum deviation allowed from the model output given the full set of features. To find the best feature selection, we divide a validation set $V \in \mathbb{R}^{n \times m_v}$ from our training data $X \in \mathbb{R}^{n \times m}$ with targets $y_v \in \mathbb{R}^{m_v}$. The goal of our feature selection will be to find the selection S that minimizes the error between the output of the full model M and the expected output of the model $v_{M,x}(S)$ for the data points in the validation set. To measure this error, we use the Root Mean Squared Error (RMSE):

$$\mathrm{RMSE}(\hat{\mathbf{y}}, \tilde{\mathbf{y}}(S)) = \sqrt{\mathrm{MSE}(\hat{\mathbf{y}}, \tilde{\mathbf{y}}(\mathbf{S}))} = \sqrt{\frac{1}{m_v} \sum_{i=1}^{m_v} (\hat{y}_i - \tilde{y}_i(S))^2}, \tag{4.12}$$

where $\hat{\mathbf{y}}$ is the output of the full model M on the validation set and $\tilde{\mathbf{y}}(S)$ is the expected

Algorithm 4: Conditional value function for all S

Input : x, a data point,
M = {val, l, r, f, t, c }, the model consisting of values, left children, right children, split feature, threshold and cover
Output: $out \in \mathbb{R}^{2^n}$, a vector storing the expected model output for every $S \in 2^P$

```
1  out = zeros(2^P)
2  weights = ones(2^P)
3  Function traverse(weights, node):
4      if node is internal then
5          setswithf = vector of length 2^P
6          setswithoutf = vector of length 2^P // see Figure 4.6
7          childrenx = next children according to x
8          childrenother = children not according to x
           /* fractions to update weights for setswithoutf:                    */
9          wx = c[childrenx] \ c[node]
10         wother = c[childrenother] \ c[node]
           /* recursive call following x:                                      */
11         traverse(setswithf + setswithoutf * wx) ⊙ weights, childrenx)
           /* recursive call not following x:                                  */
12         traverse(setswithoutf * wother) ⊙ weights, childrenother)
13     else
           /* Update output on leaf nodes:                                     */
14         out = out + weights * val[node]
15     end
16 EndFunction
17 return traverse(0)
```

answer of the model M when only the features in S are known, as it is calculated in Algorithm 4. The whole process is shown in Figure 4.5.

Runtime Analysis

The runtime of Algorithm 1 is analyzed by Lundberg et al. (2020). The complexity of a single execution of the algorithm is proportional to the number of leaves l in the respective tree. For our use case, we need to analyze every individual feature subset of the 2^P possible feature subsets. For an ensemble of k trees, we need to repeat this procedure k times, leaving us with a runtime of $O(kl2^P)$. Although this runtime is still exponential in the number of features, an algorithm for Shapley value calculation in polynomial time for random forests is stated above in this thesis. Algorithm 4 only needs one execution per tree, since every subset value is updated simultaneously. In this algorithm, the elementwise multiplication in lines 12 and 13 to update the weights is the most costly part, since all 2^P subsets are updated. The number of updates is again proportional to the number of leaves in a tree, and we need to perform this procedure for every tree in the ensemble, again leaving us with a runtime of $O(kl2^P)$. For practical applications, it is beneficial that the most expensive part of the algorithm is the elementwise multiplication of two vectors, as this is a common problem and offers multiple ways of efficient implementation, most importantly easy parallelization and the use of powerful GPUs (Okuta et al. 2017, Lam et al. 2015). It is not possible to improve the runtime of CFS by applying the same idea as for the fast calculation of the Shapley values. The main concept is to iteratively update the calculated Shapley values when traversing the tree, eliminating the need to track the value function for every existing

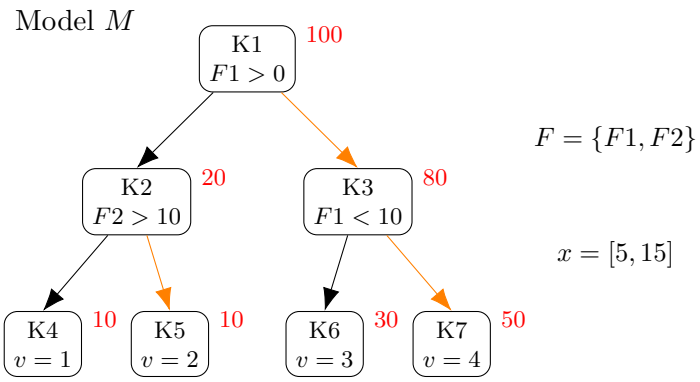


Figure 4.7: An example model for a prediction problem consisting of two features $F = \{F1, F2\}$. The first row in each node shows its name, the second shows the split condition for internal nodes and the output value for leaf nodes. Meeting the condition of an internal node means following the right-hand path. The example data point $x = [5, 15]$ follows the right path in each node, as is indicated by the orange arrows. Lastly, the red number shows the number of data points covered by each node. Figure previously published (Huber & Steinhage 2024d).

subset of players. However, for CFS, we are relying on tracking the change of every subset, making a runtime of $O(kl2^P)$ the best we can reach.

CFS Example

We use Algorithm 2 to calculate the expected output of the model defined in Figure 4.7 given that only subsets of the entire feature space F are known, for the example data point $x = [5, 15]$. For better readability, we omit the brackets for the feature sets and write $F12$ for $\{F1, F2\}$. After the first call of *traverse* we will only track the next recursive function calls for the internal nodes or the update of the output for the leaf nodes. First we initialize:

$$
\begin{array}{lcl}
\text{All subsets} & = & [\emptyset, F1, F2, F12] \\
\text{setsWithF1} & = & [0, 1, 0, 1] \\
\text{setsWithoutF1} & = & [1, 0, 1, 0] \\
\text{out} & = & [0, 0, 0, 0] \\
\text{weights} & = & [1, 1, 1, 1]
\end{array}
$$

Algorithm 2 starts with *traverse*([1,1,1,1], K1). According to the data point x, we would follow the right path in the tree towards node K3. This determines childrenx and childrenother in line 7 and 8 of Algorithm 4. The two weights are then updated according to the cover of the two nodes in lines 9 and 10. For both children we start the next recursive call of the algorithm.

split feature = $F1$
childrenx = K3
childreenother = K2
wx = 80/100
wother = 20/100
recursive call : $traverse(([0,1,0,1] + [\frac{8}{10}, 0, \frac{8}{10}, 0]) \odot [1,1,1,1], \text{K3})$ [1]
Update with full weight when F1 is known and partial weights when not
recursive call : $traverse([\frac{2}{10}, 0, \frac{2}{10}, 0] \odot [1,1,1,1], \text{K2})$ [2]
Update with zero weight when F1 is known and partial weights when not

[1] $traverse(([\frac{8}{10}, 1, \frac{8}{10}, 1]), \text{K3})$

recursive call : $traverse(([0,1,0,1] + [\frac{5}{8}, 0, \frac{5}{8}, 0]) \odot [\frac{8}{10}, 1, \frac{8}{10}, 1], \text{K7})$[3]
recursive call : $traverse([\frac{3}{8}, 0, \frac{3}{8}, 0] \odot [\frac{8}{10}, 1, \frac{8}{10}, 1], \text{K6})$ [4]

[3] $traverse(([\frac{1}{2}, 1, \frac{1}{2}, 1]), \text{K7})$

out = $[0,0,0,0] + [\frac{1}{2}, 1, \frac{1}{2}, 1] * 4 = [2,4,2,4]$
The value 4 of leaf K7 is added to the output
The weights reflect that the value is added fully for sets including F1
For sets not including F1 $\frac{1}{2}$ of training examples reach K7

[4] $traverse(([\frac{3}{10}, 0, \frac{3}{10}, 0]), \text{K6})$

out = $[4,2,4,2] + [\frac{3}{10}, 0, \frac{3}{10}, 0] * 3 = [\frac{29}{10}, 4, \frac{29}{10}, 4]$
The value 3 of leaf K6 is added to the output
The weights reflect that no value is added for sets including F1
For sets not including F1 $\frac{3}{10}$ of training examples reach K6

[2] $traverse(([\frac{2}{10}, 1, \frac{2}{10}, 1]), \text{K2})$

recursive call : $traverse(([0,0,1,1] + [\frac{1}{2}, 0, \frac{1}{2}, 0]) \odot [\frac{2}{10}, 0, \frac{2}{10}, 0], \text{K5})$ [6]
recursive call : $traverse([\frac{1}{2}, \frac{1}{2}, 0, 0] \odot [\frac{2}{10}, 0, \frac{2}{10}, 0], \text{K4})$ [7]

[6] $traverse(([\frac{1}{10}, 0, \frac{2}{2}, 0]), \text{K5})$

out = $[\frac{29}{10}, 4, \frac{29}{10}, 4] + [\frac{1}{10}, 0, \frac{2}{10}, 0] * 2 = [\frac{31}{10}, 4, \frac{33}{10}, 4]$
The value 2 of leaf K5 is added to the output
When F1 is known, K5 is never reached and no value is added

[7] $traverse(([\frac{1}{10}, 0, 0, 0]), \text{K5})$
out = $[\frac{31}{10}, 4, \frac{33}{10}, 4] + [\frac{1}{10}, 0, 0, 0] * 1 = [\frac{32}{10}, 4, \frac{33}{10}, 4]$
The value 1 of leaf K4 is added to the output
Only when F1 and F2 are unknown the path is followed to add value

The result $[\frac{32}{10}, 4, \frac{33}{10}, 4]$ can now be directly interpreted as the expected output of the model M for the data point x when we only assume access to the information in the subsets of features $[\emptyset, F1, F2, F12]$. In the case of this model, we see that the answer

when only F1 is known is the same as when both F1 and F2 are known. In general, the idea of CFS is to find the combination of features that minimizes the difference between the expected model answer for the feature subset and the output of the full model. We see that the main contribution to the runtime comes from the elementwise multiplication $\odot$. For n features the operation adds 2^n multiplications per weight update, growing exponentially in the number of features. We can now use the obtained values to calculate the Shapley values $\varphi_1(v)$ and $\varphi_2(v)$:

$$\begin{aligned}\varphi_1(v) &= \tfrac{1}{2}(4-\tfrac{33}{10})+\tfrac{1}{2}(4-\tfrac{32}{10}) = 0.75\\ \varphi_2(v) &= \tfrac{1}{2}(4-4)+\tfrac{1}{2}(\tfrac{33}{10}-\tfrac{32}{10}) = 0.05\end{aligned}$$

In this case, the Shapley values hint towards the same selection as CFS, as $\varphi_1(v)$ reveals the higher impact of feature F1 on the model output. We saw in the previous example that the evaluation of the expected model output for decision trees makes it necessary to track 2^n weights for every path in the tree. Shapley values can be calculated by only tracking $2 * n$ values for every path, since each path has a positive and a negative contribution to the calculation of the Shapley value. The contribution is positive for every occurrence of the path in the calculation of $v(S \cup i)$ in Equation (4.7). To occur in this calculation, the path must be as indicated by the example x on every split that includes the feature Fi. First, we calculate φ_{F1}. We do so by listing the positive and negative impact on the final Shapley value of every path of the model, represented by the leaf node the path ends in:

$$\begin{aligned}\text{K4:}\quad \text{pos} &= 0 & \qquad \text{K5:}\quad \text{pos} &= 0\\ \text{neg} &= \tfrac{1}{2}*\tfrac{2}{10}*\tfrac{1}{2} & \text{neg} &= \tfrac{1}{2}*\tfrac{20}{100}*\tfrac{10}{20}+\tfrac{1}{2}*\tfrac{2}{10}\\ \varphi_{F1} &= 0-\tfrac{1}{20}*1=-\tfrac{1}{20} & \varphi_{F1} &= -\tfrac{1}{20}-\tfrac{3}{20}*2=-\tfrac{7}{20}\\ \text{K6:}\quad \text{pos} &= 0 & \text{K7:}\quad \text{pos} &= 1\\ \text{neg} &= \tfrac{3}{10} & \text{neg} &= \tfrac{5}{10}\\ \varphi_{F1} &= -\tfrac{7}{20}-\tfrac{3}{10}*3=-\tfrac{25}{20} & \varphi_{F1} &= -\tfrac{25}{20}+(1-\tfrac{1}{2})*4=0.75\end{aligned}$$

And now for φ_{F2}. First, we note that the paths ending in K6 and K7 have no impact on φ_{F2}, as the feature does not appear within the paths and thus the positive and negative contributions on the Shapley value will always be equal. For the remaining paths:

$$\begin{aligned}\text{K4:}\quad \text{pos} &= 0 & \qquad \text{K5:}\quad \text{pos} &= \tfrac{1}{2}*0+\tfrac{1}{2}*\tfrac{2}{10}\\ \text{neg} &= \tfrac{1}{2}*\tfrac{1}{10}+\tfrac{1}{2}*0 & \text{neg} &= \tfrac{1}{2}*\tfrac{2}{10}*\tfrac{10}{20}+\tfrac{1}{2}*0\\ \varphi_{F2} &= 0-\tfrac{1}{20}*1-\tfrac{1}{20} & \varphi_{F2} &= -\tfrac{1}{20}+(\tfrac{1}{10}-\tfrac{1}{20})*2=0.05\end{aligned}$$

This concludes the example and shows why Shapley value calculation on random forests can be faster than performing exhaustive feature selection like CFS.

4.4 Deep Interpolation for Gaps in Remote Sensing LST Data

We want to apply our methods established in the previous three sections to yield prediction scenarios of high spatio-temporal resolution, like, for example, the prediction of grapevine yields on plot-level. For tasks like this, it is not possible to use remote sensing data out of the box, as clouds and cloud shadows produce gaps in the data. In this section, we describe three main ideas of our deep interpolation approach to interpolate gaps in remote sensing Land Surface Temperature (LST) data. First, we describe how we use individual-trained linear regressions to obtain surface temperature approximations for missing LST values when a ground-site weather station is available. Second, we describe the U-Net structure and partial convolutions that perform the interpolation of all other missing values, and third, we describe how we use a partially computed loss function for neural network training to obtain a capable model without the use of non-occluded LST representations with no missing data. We also note that throughout the whole process, we excluded each 140 LST images with full coverage for validation and testing, so they are not used during creating the linear regression models or the training of the network. The contents of this section are previously published (Huber et al. 2024b).

Linear Regression for Homogenization of Air Temperature

The first step in our efforts to interpolate holes in remote sensing LST data is to include a reliable source of information in our data. Ground-site weather stations, as they are present throughout the world, have the advantage of frequent and reliable climatic records for a specific position. Integrating this information allows us to establish a minimum amount of information for every day of the year. As the stations in our interest region only offer air temperature records, we first need to infer a surface temperature estimation from the data. We do so by training a new linear regression model for every individual ground-site weather station. It is not possible to integrate the data captured from the stations directly into the LST images, as the measured air temperature usually deviates from the surface temperature, although the two are closely correlated. To obtain data to train the linear regression models, we first search for all valid LST information regarding pixels that cover the same area as the respective ground site weather station. This leaves us with about 1000 to 1600 entries depending on the weather station. An outlier is the Freimersheim station with only 438 valid entries in the training data because it is available only from January 1, 2019. For each station, we choose to use 80% of the data for training and 20% of the data to validate and adjust the process. Note again that the final testing data are excluded throughout the process and the interpolated LST values are part of calculating the final accuracies of all the approaches tested. Regarding the input features to our models, as described in Section 3.2, we use the average, minimum, and maximum air temperature, together with the air humidity captured at 6:00 am and 11:00 am, to infer the surface temperature at 11:00 am clock of the pertaining day. All features are measured twice, once at a height of 20 cm and once at a height of 200 cm. The RMSE to homogenize the air temperature with LST is around 2.5 °C for each station, with the correlation coefficient R2 around 0.9. An explanation for both evaluation metrics can be found at the beginning of Chapter 5 and an exact listing of the results is shown in Table 4.1. Experiments with more

advanced machine learning models like random forests or XGBoost have not shown any improvements, resulting in us choosing linear regression, due to its fast training and evaluation together with high robustness.

Name	Latitude	Longitude	RMSE (°C)	R2
Bad Bergzabern	49.11	8.00	2.80	0.92
Deidesheim Niederkirchen	49.43	8.22	2.70	0.94
Edesheim	49.26	8.15	2.82	0.94
Ellerstadt	49.46	8.27	2.47	0.95
Freimersheim	49.27	8.22	2.70	0.92
Goecklingen Holzbruehl	49.16	8.03	2.64	0.93
Herxheimweyher	49.16	8.25	2.88	0.92
Lachen Speyerdorf	49.31	8.20	2.66	0.94
Landau Nussdorf	49.22	8.11	2.75	0.94
Landau Wollmesheim	49.18	8.08	2.60	0.94
Maikammer	49.29	8.15	2.72	0.94
Meckenheim	49.40	8.26	2.77	0.94
Neustadt an der Weinstraße	49.37	8.19	3.07	0.91
Ruppertsberg	49.39	8.19	2.77	0.93
Schaidt	49.05	8.09	2.30	0.93
Schweigen Rechtenbach	49.05	7.97	2.71	0.93
Schweighofen	49.04	7.99	2.48	0.93
Siebeldingen	49.22	8.05	2.67	0.93
Steinweiler	49.10	8.10	2.69	0.93
Wachenheim	49.44	8.19	2.84	0.93

Table 4.1: Overview over the location of all 20 local ground-site weather stations used to obtain air temperature data. The values of RMSE and R^2 for the linear regression models that convert the air temperature into LST data. Table previously published (Huber et al. 2024b)

Partial Convolutions for Interpolating Temperature

We propose a U-Net-based architecture inspired by the work of Ronneberger et al. (2015) to fill the gaps in the LST data. The general structure of the network is shown in Figure 4.8. The input and output of the image consist of an LST image measuring 69x58 pixels. At each level of the U-Net architecture, we perform two concatenated 5x5 partial convolutions to extract features, shown by black arrows in Figure 4.8. Partial convolutions, as introduced by Liu et al. (2018), work the same as regular convolutions except that they do not include masked pixels. Given convolution filter weights W, the input features for the current convolution window are written as X. M is the binary mask that indicates missing values in X. The bias is b and the constant maximum number of possible valid pixels in one convolution step is written as c. For each X' of the output, a partial convolution can be described as:

$$X' = \begin{cases} W^T(X \odot M)\frac{c}{sum(M)} + b, & \text{if } sum(M) > 0, \\ 0, & otherwise. \end{cases} \tag{4.13}$$

Here, $\odot$ is the elementwise multiplication. The fraction in this equation can be inter-

preted as a scaling factor for the number of valid pixels in the calculation. The condition $sum(M) > 0$ checks if at least one valid pixel is available. The mask is updated under the same conditions. If $sum(M) > 0$, then M', the new mask, gets a valid pixel at that position. After the first set of convolutions, the downward path starts with a combination of a leaky Rectified Linear Unit (ReLU) and a 2x2 max pooling layer. Leaky ReLU is an activation function similar to a standard ReLU activation. With a standard ReLU activation, all negative values are suppressed, and positive values are propagated linearly with a slope of 1. A leaky ReLU treats positive values as the same, but negative values are not suppressed but linearly discounted. The max-pooling step selects the maximum value from a 2x2 window. On the one hand, this helps to select the most important extracted features in each step, and on the other hand, this helps us to interpolate missing values, as one valid value can be selected to represent up to three missing values. The application of a leaky ReLU and the max pooling is shown by blue arrows in Figure 4.8. The lowest layer of our U-Net is called the bottleneck and consists of image tensors of the size 4x3. The small size guarantees that after an empty LST image is filled with the up to 20 valid pixel values extracted from ground-site weather stations, there are no more missing values in the bottleneck and therefore there are no missing values throughout the whole upward path and subsequently no missing values in the final output image. The stepwise increase in image size in the upward path is carried out by the so-called inverse convolutions (Zeiler et al. 2010) with a 2x2 kernel and a stride of 2, shown as orange arrows in Figure 4.8. Lastly, at every step of the upward path, we copy the state of the last output of the same layer of the downward path and concatenate the input to the first convolution of the upward path. This is also standard practice for designing U-Net structures and helps stabilize the training process. Finally, the network output is smoothed with a Gaussian kernel of size 5 with a standard deviation of 0.7. This further improves the results, as LST images are mostly rather smooth, while the network output can have steep differences between neighboring pixels.

Learning on Occluded Data

As fully available ground-truth data for LST reconstruction are very limited, we need a solution to train our network using partially occluded ground-truth data. This means that we have input data where only a portion of all pixels are valid. To create pairs of input and output data that can be used to train the network, we removed additional pixel information from each image. To create training images that resemble the real-world conditions the best, we take the pattern of occluded pixels (cloud mask) from a random different image of our training data set. This often results in some pixels being masked, where ground-truth data are available, which can ultimately be used to estimate how well the network can fill gaps in LST data. An overview of the process is shown in Figure 4.9. The average amount of non-valid pixels in the training images is increased throughout this procedure from 2505 non-valid pixels to 2729 non-valid pixels. The increase in those numbers is to be expected, as the amount of non-valid pixels per image can only be increased for every individual image. Throughout this process, we also create training images where no additional pixels are made invalid, resulting in the output image not having any additional information compared to the input images. However, including those images in training the network is beneficial as

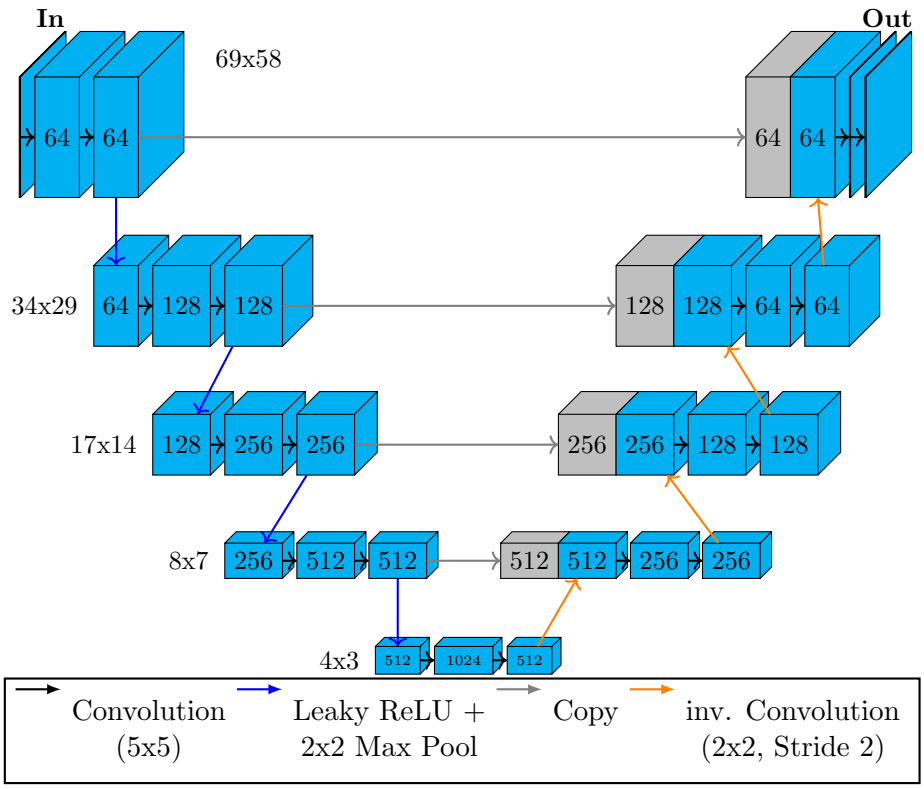


Figure 4.8: Visualization of the U-Net structure including partial convolutions used to interpolate gaps in LST data. Figure previously published (Huber et al. 2024b).

the network can use those to learn the spatial dependencies of the LST images in our study region and therefore further improve accuracy and stabilize training. During the training process, we used the Mean-Squared-Error (MSE) loss between the ground-truth y and the prediction $\hat{y}$: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||$. To overcome the problem of missing ground-truth data for an entire image, the MSE is calculated only where the ground-truth has viable information. We note that this includes both pixels that exist in the network input and pixels that are occluded in the network input image, providing stability during the training process. For the final evaluation of the test data, only pixels newly inferred by the network are used to calculate the error metrics.

4.5 Regularized Deep Transfer Learning for Yield Prediction

To conclude the chapter, we will explain the methods used to achieve knowledge transfer between two different yield prediction domains. For this, we use multiple different tools to achieve the best accuracy for yield prediction on a smaller data set. To allow for successful transfer learning, we align our two data sources first, before we apply multiple regularization techniques, and subsequently append a Gaussian process to recover the information lost due to histogramization. An overview of the process is shown in Figure 4.10. The basis for our deep learning model is the same as that used to compare our results in Section 4.1. For transfer learning, we use deep learning instead of XGBoost,

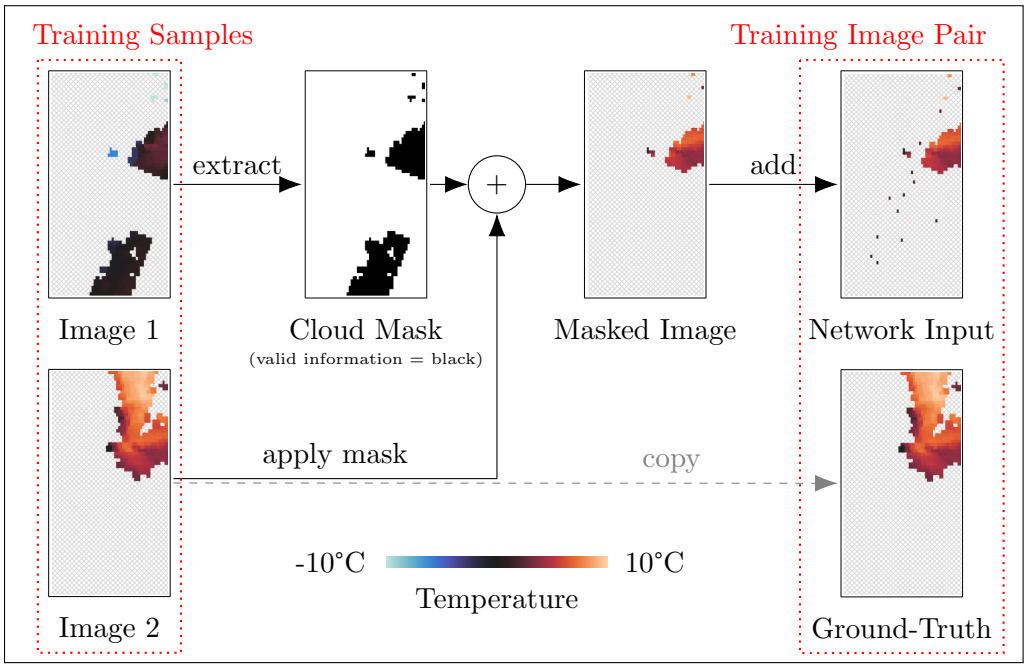


Figure 4.9: Overview of creating image pairs for network training. Two random training images are selected, one for extracting a real-world example for missing values and one for ground-truth temperature data. The second image is masked according to the missing values in image one. The network input is then completed by adding LST data, where weather stations are available. Figure previously published (Huber et al. 2024b)

as decision tree approaches to transfer learning (Segev et al. 2016, Jiang et al. 2019) are not as powerful, especially when the respective domains are inhomogeneous.

Spatio-temporal Alignment

The geographical distance between the US and Argentina accounts for some differences in soybean cultivation in both countries. Specifically, we need to adjust for different crop growth cycles. In the US, we monitor the growing conditions from March 23 to December 4. In Argentina, due to different climatic conditions, we monitor the crop growth cycle from November 26 to August 9. Both periods extend beyond harvest time, so that we can leverage the capabilities of our models to determine important input data themselves. As soybean yield prediction is especially valuable during the early stages of soybean growth, we examine a second in-season forecast to test our models. For this case, the time period only contains the first 14 of the 34 8-day intervals. The time periods used, as well as the underlying crop calendars (USDA 2023), can be seen in Figure 4.11. After data cleansing, a histogramization is performed, as first done by You et al. (2017) and previously explained in Section 4.1. For each band and each satellite image, the pixels masked as cropland within a county are summed for each record. Subsequently, these sums are transformed into a normalized histogram with 32 bins of the frequency distributions of the records in the vertical and the associated time in the horizontal. The resulting data point consists of 9 times 34 histograms, a county name and a season year.

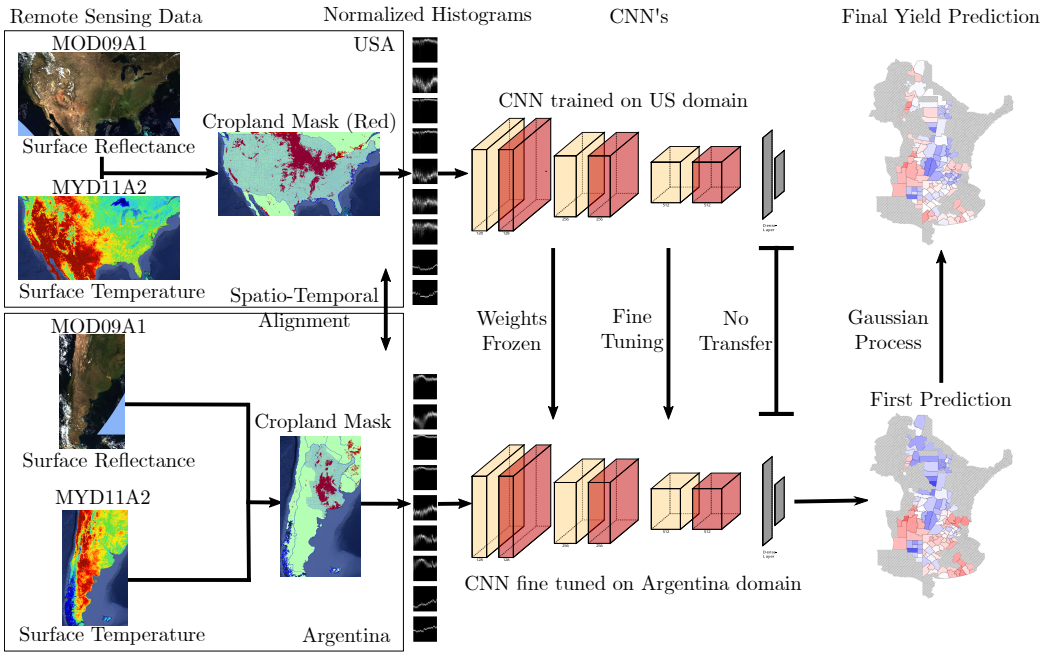


Figure 4.10: Overview of our approach for leveraging remote sensing data for yield prediction with deep transfer learning. The two boxes on the left side show the kind of input data that we use, consisting of surface reflectance data, surface temperature data, and the cropland mask for each of the two domains. The data is spatio-temporally aligned as preparation for transfer learning, before being processed toward normalized histograms. The middle section of the figure shows some of the transfer learning techniques used: frozen weights, fine-tuning with regularization, and the initialization of the dense layer at the end of the network. Lastly, on the right-hand side, we show an example of refining the prediction results by applying a Gaussian process. Figure previously published (Huber et al. 2024a).

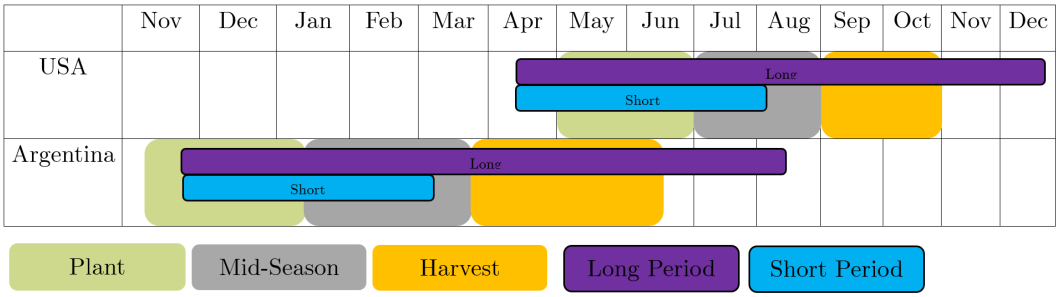


Figure 4.11: Crop calendar of Argentina and USA (data from USDA (2023)). The data of the long period is shown in purple. The data of the short period is shown in blue. Figure previously published (Huber et al. 2024a).

Model Design and Transfer Learning Techniques

At the center of our predictive model is a CNN, as it is first used for yield prediction by You et al. (2017). As a baseline for our implementation, we started from the work of Tseng (2022). The CNN consists of 6 convolutional layers and the following dense layer. As a result of histogramization of the input data, we need an alternative solution to the commonly used pooling layers in conjunction with the convolutional layers. The solution is convolutional layers with a stride of 2 to prevent locational invariance due to pooling. In addition, we use dropout layers, early stopping, and batch normalization when training the CNN. Foremost, to apply transfer learning, this CNN is trained in the US domain and will be referred to as the US model in the following. The most natural way to apply transfer learning is to fine-tune the US model in the Argentine domain. This corresponds to initializing the weights of the CNN for the Argentine domain with the weights of the US model. By assuming a similar distribution of the US domain and the Argentine domain, the starting point of the Argentine model will thus be chosen closer to the desired model, and the hypothesis space will be constrained by limiting the hyperparameters of the training. Furthermore, since it is known through extensive research that later layers have greater domain-specific significance while early layers extract general properties, it may be useful to distinguish the approach based on layer depth (Plested & Gedeon 2022). The first way to treat layers differently is to freeze some of them. The frozen layers keep their weights unchanged during fine-tuning and are effectively not trained directly on the new domain. In our case, this means that the frozen layers are trained only on the larger US domain to contribute to a prediction on the Argentine domain. Due to the high generalization of the front layers, this procedure is applied to the front convolutional layers and thus adopts the basic framework of the US model for feature extraction. This serves, in particular, to avoid catastrophic forgetting (Iman et al. 2022). Catastrophic forgetting is one of the two major problems in transfer learning and is the tendency of neural networks to abruptly lose the knowledge of previous tasks when learning a new task (Kirkpatrick et al. 2017). Later convolutional layers are initialized with the weights of the US model, but retain their ability to adapt to the Argentine domain. To name this retraining on top of the source domain, we overload the term fine-tuning, which also refers to the complete retraining of the net on the Argentine domain on top of the US model. Last, we use the ability to completely retrain layers independent of the US model so that they can fully adapt to the Argentine domain. This complete retraining is used in the dense layers to account for their strong specialization to the target domain associated with their high depth.

Transfer-specific regularization methods are suitable to further limit catastrophic forgetting and to avoid negative transfer. Regularization influences CNN training by an additional summand Ω to the loss. The widely known $\mathbf{L^2}$ regularization, also known as weight decay, penalizes large weights in the CNN. In this process, as seen in Equation (4.14), the vector w of weights of the CNN is normalized, squared, and parameterized by α to set the regularization strength:

$$\Omega(w) = \frac{\alpha}{2}\|w\|_2^2. \tag{4.14}$$

Based on this, Xuhong et al. (2018) modifies the starting point of this regularization so

that instead of a deviation of the weights from zero, a deviation of the weights from the US model is penalized. Thus, the hypothesis space is again restricted to the surroundings of the US model, making catastrophic forgetting less likely. Since for the deviation of the target model from the source model an equal structure of both networks is required, Xuhong et al. (2018) extends the model by the option to regularize non-transferable and especially newly added parts of the model with the $\mathbf{L^2}$ regularization. Equation (4.15) shows the corresponding formula, where α and β set the strengths of the regularization of the constant and newly added parts:

$$\Omega(w) = \frac{\alpha}{2}\|w_S - w_s^0\|_2^2 + \frac{\beta}{2}\|w_{\bar{S}}\|_2^2, \tag{4.15}$$

where w_S is a weight vector of the constant structures of the target model, w_s^0 is a weight vector of the constant structures of the source model and $w_{\bar{S}}$ is the weight vector of the newly added structures of the target model. We exploit the application of $\mathbf{L^2}$ regularization for greater adaptability of the dense layer, so that dense layers are effectively regularized with $\mathbf{L^2}$ regularization even when $\mathbf{L^2}$-SP regularization is applied. This means in particular that the vectors of the constant structures are equal to the convolutive layers and the newly added structures are equal to the complete reinitialization of the dense layers.

Negative transfer refers to a loss of performance due to knowledge transfer and occurs due to a lack of transferability of some features caused by differences in the domains (Chen et al. 2019, Plested & Gedeon 2022). To reduce negative transfer, we use the Batch Spectral Shrinkage (BSS) regularization method. Chen et al. (2019) observed that large data sets lead to highly generalized models, while at the same time these models suppress small singular values of the extracted features. To exploit this behavior of generalized models, Chen et al. (2019) use artificial suppression of small singular values. Specifically, they introduced the concept of relative angles between domains that can be used to measure the transferability of eigenvectors in the weight matrices. Subsequently they found out that in later layers of the network only eigenvectors corresponding to relatively large eigenvalues produce small angles and hence are well suited for knowledge transfer. Here, the feature maps f_i of the input x_i are vectorized and aggregated into a feature matrix $F = [f_1 ... f_b]$ per batch of size b. Subsequently, the singular value decomposition is applied to this feature matrix, and the k smallest singular values are used to suppress the associated poorly transferable eigenvectors. The BSS regularization is thus given by

$$\Omega(F) = L_{bss}(F) = \eta \sum_{i=1}^{k} \sigma_{-i}^2,$$

where η as a hyperparameter specifies the strength of the regularization, k specifies the number of smallest singular values σ_{-i} to be penalized. k is set to 1 according to Chen et al. (2019) and the regularization is completely adjusted through η. The use of BSS and $\mathbf{L^2}$-SP have the advantage that both regularization methods can be applied simultaneously (Chen et al. 2019).

Deep Gaussian Process

To extend the CNN to include spatio-temporal features, You et al. (2017) propose the use of a Gaussian process on top of the CNN. For a data point x, the deep Gaussian process uses the input of the last dense layer of the CNN $h(x)$ and a Gaussian process $f(x) \sim \mathcal{GP}(0, k(x, x'))$ to make a harvest prediction:

$$y(x) = f(x) + h(x)^T \beta$$

$f(x) \sim \mathcal{GP}(0, k(x, x'))$ is distributed from a Gaussian process with zero mean and covariance defined by the squared exponential kernel

$$k(x, x') = \sigma^2 \exp\left(\frac{\|g_{loc} - g'_{loc}\|_2^2}{2r_{loc}^2} - \frac{\|g_{year} - g_{year'}\|_2^2}{2r_{year}^2}\right) \sigma_e^2 \delta_{g,g'}$$

where g_{loc} and g_{year} indicate the spatial and temporal data of the associated data point x and $\delta_{g,g'}$ is the Kronecker delta as noise factor over $g = (g_{loc}, g_{year})$ parameterized by σ_e. σ, σ_e, r_{loc} and r_{year} are hyperparameters. In summary, $f(x)$ distributed from a Gaussian process is a collection of random variables with a joint Gaussian distribution. The next summand for the prediction is given by a set of basis functions $h(\cdot)$, corresponding to the last layer, and a random variable β. β is distributed from a normal distribution with the weight vector of the last layer as mean and $\sigma_b I$ as the variance with the hyperparameter σ_b.

The choice of β and $h(x)$ results in a distribution around the CNN prediction. The second summand then results from the covariance of the training data with the test data formed on the RBF kernel multiplied by the differences of the prediction $h(x)^T \beta$ on the training data and the ground-truths of the training data, which are again weighted with the help of their covariance. This forms $f(x) \sim \mathcal{GP}(0, k(x, x'))$, which reduces the error values depending on the local and temporal relationships of the underlying data.

5 Experiments and Discussion

Within this chapter we present and discuss our experiments that analyze the impact of our solutions on the five challenges that we face when predicting agricultural yields with machine learning: training on small data sets, need for explanations, selecting important features, gaps in remote sensing data, and making predictions for shifting domains. We start by evaluating our pipeline processing remote sensing data into a tabular representation by estimating the underlying distributions to allow predictions with XGBoost by comparing the prediction for soybean yields in the US against the state-of-the-art deep learning approaches in Section 5.1. In Section 5.2 we explain the resulting model using grouped Shapley values and show how our models decision process is in line with the expertise in the field of yield prediction. We then have an application-driven discussion on the use of Shapley values for feature selection in Section 5.3 by comparing different selection methods in multiple real-world examples, before using the Shapley values to select an expressive subset of features to optimize the data efficiency of our previously built yield prediction model. Furthermore, we test the capabilities of our deep interpolation approach for remote sensing LST data to prepare data for the prediction of grapevine yield in Section 5.4. Lastly, we analyze transfer learning efforts to transfer knowledge from soybean yield prediction in the US to soybean yield prediction in Argentina in Section 5.5.

Performance Metrics

Throughout this section, we will use two main metrics to evaluate our approaches. To measure the overall error, the Root Mean Square Error (RMSE) is calculated as follows:

$$\mathrm{RMSE}(\hat{\mathbf{y}}, \mathbf{y}) = \sqrt{\mathrm{MSE}(\hat{\mathbf{y}}, \mathbf{y})} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}. \tag{5.1}$$

Here, $\mathbf{y}$ represents the ground-truth values and $\hat{\mathbf{y}}$, the prediction values. To be able to measure the variability in the target variable that is explained by the models, the coefficient of determination (R^2) is utilized:

$$R^2(\hat{\mathbf{y}}, \mathbf{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{5.2}$$

where $\bar{y}$ is the mean of the ground-truth values $\mathbf{y}$.

Furthermore, every experiment that includes random aspects consists of multiple runs. This includes accounting for randomness in the initialization of deep learning frameworks and the sub-sampling within the XGBoost algorithm.

5.1 Comparative Evaluation of XGBoost vs. Deep Learning for Yield Prediction

The performance of the XGBoost-based approach for yield prediction and the deep learning approaches are evaluated in two different yield prediction scenarios. An end-of-the-year prediction and an in-year prediction on soybean yields provided by the USDA. The experiments have previously been published (Huber et al. 2022).

End-of-the-Year Prediction

Experiments are carried out on the entire data set to evaluate all approaches for an end-of-year yield prediction. This means that the complete span between the 49th and 321st day of the year is used. By this date, all soybeans should be harvested (USDA 2010), and therefore the satellite data include information on the farmland right before harvest. When testing approaches for a specific year of the data set, it is assumed that only data from before the specific year are available for training and validation, while the year itself is explicitly excluded for testing. Hyperparametertuning was run for 50 iterations without further pruning involved to optimize the accuracy on the validation set. A model is tuned and trained for each testing year from 2017 to 2021, using 10% of the training data available for validation. After being used for validation, the data is included again in the training set for the final model. A unique model is trained for each year to be as close as possible to the real-world use case, where it should be beneficial to include the most recent years in training. The results can be observed in Table 5.1. When comparing the XGBoost-based approach with deep learning approaches, RMSE decreases by about 25% when averaging the metric over five years in the test data. In the same time frame, the average R^2 score increases by about 0.13. The good performance of the XGBoost-based approach in the experiments can be explained by the nature of the data. Although remote sensing data are often referred to as satellite images, there is a greater resemblance to tabular data sets, where the information is represented by numerical features instead of pixels. The necessary preprocessing towards histograms or distribution approximation values strengthens this supposition. Classic machine learning approaches on feature-engineered data sets thrive under these conditions.

With respect to the results of hyperparameter tuning, a tendency of the models to develop deep trees is prevalent. The parameter that provides the maximum allowed depth of the individual trees averages 21.8 over all five years in the experiments. The XGBoost library defaults towards a value of 6 for this specific parameter, since the standard use cases contain smaller quantities of overall features. This indicates that the data at hand are of high complexity and need many features to be included for an informed regression. Since this could lead to overfitting, other hyperparameters are tuned to encourage conservative model behavior. The learning rate is rather low and averages 0.06 compared to the default value of 0.3. Furthermore, the hyperparameters indicating stochastic sampling of rows and columns in each iteration average at 0.6 and 0.84 respectively, to help avoid overfitting.

	XGBoost		CNN		CNN-LSTM	
Year	RMSE	R^2	RMSE	R^2	RMSE	R^2
2017	3.77	0.82	5.01	0.70	5.07	0.70
2018	4.51	0.76	6.15	0.63	6.21	0.63
2019	4.21	0.76	5.52	0.57	5.76	0.54
2020	4.22	0.80	6.66	0.55	7.63	0.42
2021	4.55	0.82	5.12	0.85	6.03	0.79
AVG	4.25	0.79	5.69	0.66	6.14	0.61

Table 5.1: RMSE and R^2 values of end-of-year soybean yield predictions. The best result for every year is highlighted in blue. The CNN architecture is taken from You et al. (2017), and the CNN-LSTM architecture is taken from Sun et al. (2019). Table previously published (Huber et al. 2022).

	XGBoost		CNN		CNN-LSTM	
Year	RMSE	R^2	RMSE	R^2	RMSE	R^2
2017	4.88	0.72	7.82	0.28	7.78	0.29
2018	5.27	0.68	7.30	0.50	8.06	0.38
2019	4.65	0.70	9.73	-0.49	9.67	-0.31
2020	5.35	0.65	8.24	0.32	8.16	0.33
2021	6.04	0.61	8.11	0.62	8.76	0.56
AVG	5.24	0.67	8.24	0.25	8.49	0.25

Table 5.2: RMSE and R^2 values of in-year soybean yield predictions. The best result for every year is highlighted in blue. The CNN architecture is taken from You et al. (2017), and the CNN-LSTM architecture is taken from Sun et al. (2019). Table previously published (Huber et al. 2022).

In-Year Prediction

To showcase the dynamics of in-year soybean yield estimation, the time between the 49th and the 201st day of the year is used to produce training data. This ensures that the harvesting season has not yet begun. Therefore, the data contain 19 of the 8-day intervals that present the remote sensing data. The reduced amount of information increases the difficulty of the prediction task. On the same note, information gained from an in-year prediction is even more valuable to every party involved, since classic estimation approaches often include counting specific features of the crop that might not have developed yet. Training and testing data are handled as explained for the end-of-year prediction.

The results are presented in Table 5.2. Averaging error metrics over five testing years, the XGBoost-based approach outperforms the best state-of-the-art approach by 36% in terms of RMSE and 0.42 in terms of R^2. The gap between the deep learning approaches and the XGBoost approach appears to be wider than for the in-year prediction. This indicates that the XGBoost based method is less affected by the lack of information about the later growing stages, possibly hinting that the underlying relations between early growing stages and final yield are better understood than by the deep learning methods. Regarding the hyperparameters of XGBoost, similar values are observed as described for the end-of-year prediction.

Data and Time Efficiency

While the raw satellite images take up 218 GB, histogramization results in a reduction of down to 1.41 GB by removing spatial dependencies of the data. The histograms still cover the complete ranges of values of the pixel intensities. However, since adjacent image pixels have a high probability of showing similar values, multiple bins of these histograms show zero entries. Our approach of estimating only three key values, that is, the median, 20% quantile, and 80% quantile, describing the value distributions produces a representation size of 0.01 GB for further processing.

The USDA yield data set shows the most significant amount of training data that needs to be processed for the year 2021. To examine the time used to train and test a yield prediction model for 2021, the performance is averaged over two runs, since the early stopping mechanic within the training of the deep learning approaches is responsible for varying training times. The CNN network requires approximately 15 minutes for training. The CNN-LSTM architecture takes approximately 10 minutes for the task. Finally, the XGBoost-based approach takes about 90 seconds for the same task. All experiments were performed on an NVIDIA GeForce GTX 1660 Ti.

Discussion

The experiments above hint towards a good compatibility between XGBoost and the estimation of yield, showing a performance at least on par compared to deep learning for the end-of-year as well as for the in-year predictions. However, it is generally known that deep learning methods benefit from extensive training data. This tendency is especially noticeable in the CNN approach, where we see an increase in performance after expanding the training data set after early experiments. Although the XGBoost method showed similar results for the reduced and full data sets, the accuracy of the CNN improved. Another indicator of this hypothesis is the good performance of the CNN approach for yield prediction in 2021, where most data are available. At the same time, 2021 is the only year in which CNN could outperform XGBoost in terms of R^2. Although this is taken into account, the limitations of real-world data need to be acknowledged. Arguably, soybean production in the US is one of the largest data sets available to train and test models for yield prediction. Therefore, using the same approaches for other data sets will often result in fewer data available to train the models. As more years pass, it will be possible to train the models on even larger data sets with more harvest data available. In addition, other possibilities for handling scarce data in deep learning could bring improvements. For classic image processing tasks, pre-trained networks are utilized. The networks are pre-trained to extract general features on large data sets in a task agnostic way. For a specific use case, pre-trained networks are trained further on often small task-dependent use cases, achieving good accuracies on scarce data sets (Kolesnikov et al. 2020). We will provide experiments on this matter later in this thesis. Other research fields include semi-supervised learning, where only parts of the data are annotated (Pham et al. 2021), and synthetic data sets, where more training data are generated (Xu et al. 2019).

Another point worth highlighting in this study is the application of different processing methods for the same data. Experiments have shown that the respective way of

processing data, which means histograms for deep learning and features for XGBoost, obtains the highest accuracies, respectively. The difference in accuracies between the CNN and CNN-LSTM approaches must be explained. Sun et al. (2019) claim that the CNN-LSTM approach outperforms a CNN in their article. However, the CNN used for comparison consists only of the CNN part of the CNN-LSTM and is not geared specifically towards the problem. Reproducing their experiments showed the same results for the same CNN, but the CNN developed by You et al. (2017) is still able to outperform the CNN-LSTM on our data.

Furthermore, the use of vegetation indices could bring about further improvements in accuracy. Multiple studies are conducted in which remote sensing data are accessed for yield prediction, based on the results of different vegetation indices. In this study, the information integrated within the vegetation indices is expected to be incorporated directly into the models. This is due to the assumed capabilities of all the approaches involved in our study to understand the relationships between the raw remote sensing data and vegetation indices.

The results of the experiments prove the capabilities of XGBoost for soybean yield prediction in the US, compared to state-of-the-art deep learning approaches. However, this is only one possible task to predict yields out of many. Other tasks will most likely suffer from fewer available training data and will make it more difficult for deep learning approaches to reach the accuracies of our XGBoost pipeline. Selection of crops within the US allows the use of CDL and Daymet data, which are not available worldwide. Although substitutable data sources are available around the world, the quality of the data and the resulting prediction accuracy may differ, as we will see later in this thesis in Section 5.5. Not only the region, but also the crop of interest could be different for further yield prediction scenarios. We expect the results for other crops and different yield prediction scenarios, such as, for example, corn and wheat, to be similar. Finally, the limitations of XGBoost need to be acknowledged. Although very good prediction accuracies have been achieved in many real-world scenarios, some weaknesses are well known. This includes a lack of extrapolation capabilities when dealing with test data that exceed the observed feature quantities during training. Especially in times of climate change, this could pose future problems for our approach. Furthermore, XGBoost can be heavily affected by outliers due to the nature of Gradient Boosting, where each learner tries to consider the previous learners' mistakes. Since data acquisition of harvested yields always includes human-made measurements, mistakes and therefore outliers could occur.

5.2 Explaining Yield Predictions with Grouped Shapley Values

In this section, we use our novel method to calculate grouped Shapley values for random forests for the models created in Section 5.1. We show how grouped Shapley values can be used to analyze the prediction model and make the results plausible for practical use cases by showing alignment with the knowledge of experts in yield prediction by grouping the explanations first by remote sensing band and then by the timestep used for creating the features in our data. Some of the results in this section have previously been published (Huber et al. 2023).

Grouping Feature Attributions by Band

To explain our yield prediction models, grouped Shapley values are necessary, as the input data originally consists of 1131 input features, making any effort to analyze individual feature attributions tedious. We will focus our analysis on the model created for the full out-of-year prediction model, as the explanations of a model are worth more if the model is more precise. Furthermore, we will exemplarily analyze the model to predict the 2021 yields, as our experiments reveal similar plots for the prediction of any year. The input features are extracted from 11 different bands of remote sensing data with three values representing the band for a county in an eight day interval. Figure 5.1 shows the swarm plots resulting from the grouping of features with respect to their spectral bands. We note that the sum of all GSV of a local explanation will result in the model output for the specific data point. Therefore, each grouped Shapley value describes how knowing the data point's values for the features within the groups impacted the model output, in comparison to the absence of these values. If we look, for example, at the values on the far right within the swarm plot for the handcrafted features in Figure 5.1, we know that there are data points that we would expect to have ca. 15 bu/ac lower predicted yield, when the handcrafted features would be unknown. Furthermore, we know that this behavior is caused by the high overall values of the feature group, since the respective data points are colored red. When iterating through the swarm plots, we see interesting and insightful patterns in most of them. The following insights can be gained from the corresponding swarm plots:

Red - 620-670 nm and NIR - 841-876: These two feature groups are components of the well-known Normalized Difference Vegetation Index (NDVI) (Quarmby et al. 1993), which is historically used to summarize remote sensing images to predict yields. The NDVI is calculated by dividing the difference between the NIR and the Red band by their sum. This means that a high NIR will increase the NDVI and is correlated with a higher yield, while the opposite holds for the red band. This coincides with the plots in Figure 5.1, as can be seen by the reverse order of the red and blue points in the two plots. For the red band, the blue dots are mostly on the positive side of the plot, meaning that a low value for this band coincides with an increasing yield prediction. For the NIR band, we observe the opposite. The dots colored blue make the biggest negative impact on the model of all feature groups, by reducing the model's prediction by more than 5 bu/ac.

Blue - 459-479 nm and Green - 545-564 nm: Both bands have historically not been used for yield prediction and also have little impact on our prediction models, as there are no dots within the plots that show a high GSV.

NIR - 1230-1250 nm and IR - 1628-1652 nm: These feature groups show lower impacts, indicated by very narrow swarm plots. However, for both bands, we see a tendency that higher values coincide with lower predictions, since the negative impacts on the yield predictions are all recorded for red-colored dots.

IR - 2105-2155 nm: This feature group shows a higher impact, indicated by a larger swarm plot. Since we find red and blue dots at both ends of the spectrum, we cannot derive a pattern or interpret the influence of higher or lower values. This means that the model's interpretation of this feature group is highly influenced by other feature groups around, but still important to derive the final prediction.

TempDay and TempNight: The temperature at day shows a larger swarm plot than the temperature at night and therefore is more influential on the model output. At the same time, the dots within the night temperature are clearly sorted from blue to red and indicate that a higher temperature at night coincides with a higher yield prediction.

Precipitation and Vapor Pressure: The precipitation group shows almost no impact on the model output, as it is the narrowest swarm plot in Figure 4.3. The vapor pressure group shows a small impact on the yield prediction and a very clear indication that a higher vapor pressure should lead to slightly higher yields because the dots are completely in order from blue to red. Interestingly, both feature groups are not available worldwide, as they are specifically captured within the US. The relatively low impact on the prediction model encourages soybean yield prediction experiments in other regions of the world, although this information would not be available.

Handcrafted Features: This is the most influential group of features, capable of altering the prediction of yield by more than 15 bu/ac. The impact is very high, as it includes the average yield of the county represented by the data point over the foregoing years. That is, a county with traditionally higher yields in the past will obtain a higher yield prediction from our model, thus inducing spatial context in the modeling.

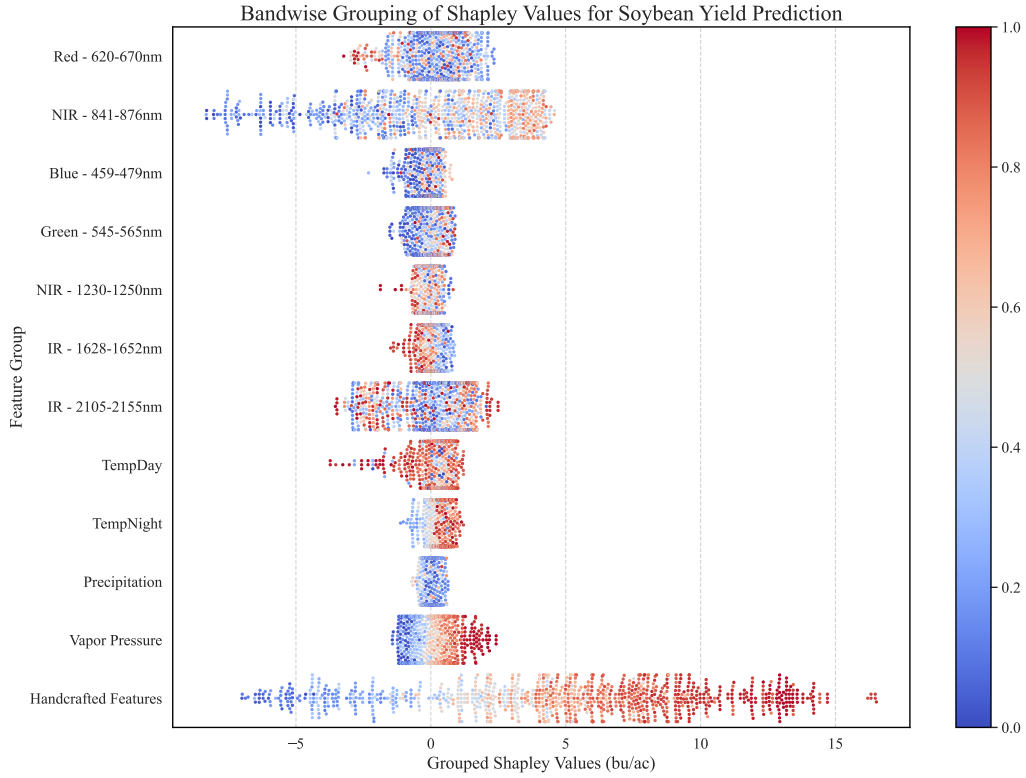


Figure 5.1: GSV feature importances for soybean yield prediction. Every dot is a local explanation of a soybean yield prediction in the U.S. The first 11 plots represent a group of 102 features each calculated from different remote sensing data. The bottom plot shows a group of 7 additional handcrafted features. The color represents an averaged and normalized representation of the grouped features values. Figure already published in (Huber et al. 2023).

Grouping Feature Attributions by Timestamp

Another natural way to group the features is not by band, but by the time step (TS) in which they are captured. For this, we leave out the handcrafted features, as they cannot be attributed to a specific time. The swarmplot for the grouped Shapley values showing the impact of each time step observations on the prediction result is shown in Figure 5.2. For this grouping, the hue of the individual points is difficult to interpret, as the aggregated value of the value of all the bands at an individual point in time is not as expressive as for the grouping by band seen in the previous experiment. However, we see an increasing importance of time steps starting from TS 17, where we see the modeling results being altered by circa -1 to +1 bu/ac by the features values. The greatest impact on the model output can be found by the features captured at TS 22 and TS 23, where unfavorable conditions can decrease the prediction of the model by up to 6 bu/ac, each. This is supported by the knowledge of experts on the prediction of soybean yields, stating that high temperature stress during mid-reproductive growth is detrimental to yields, while stress experienced during the early or late stages of development has comparatively little impact on plants (Yang et al. 2023). The figure also helps explain the differences in accuracy between the in-year and end-of-year predictions, with the in-year prediction not including the very impactful TS 22 and TS 23.

Discussion

The presented work on using GSV to gain global understanding of random forests shows several advantages over other explanation methods. Grouping similar features allows for a representation of the models decision process that can be easily understood and can serve as a tool to understand that our models decision making process is in line with experts knowledge regarding the topic and can be helpful in creating synergy between computer scientists and domain experts. Much more so than a direct representation of the importance of all the features, as can be done with classical Shapley values. One common solution to this problem prior to our work was to add up the individual Shapley value feature attributions, ignoring interactions between the features within the same group that can alter the results, as we have seen in the example in Section 4.3. Our approach does all this while conserving the common strengths of Shapley values in machine learning, offering a unified framework for explanations across different machine learning models, and most importantly preserving the axiom of efficiency, resulting in all the explanations of a data point adding up to the final model output. This makes feature attribution easy to understand, especially outside of the field of computer science. For tree models, all this is possible without having to sacrifice runtime, as we provide a fast algorithm to calculate grouped Shapley values. We acknowledge that our approach relies on a natural grouping of features that cannot be determined in every scenario but is common for yield prediction tasks, as multiple features are often derived from the same sensor or captured at the same time frames. In some cases, the restricting rules on creating the partition of predefined coalitions can hinder the analysis of important features. However, when we want to compare to predefined coalitions that share a set of features, we can use our approach multiple times to support the analysis. The easiest way would be to perform two evaluations of GSV with two feature groups each. One

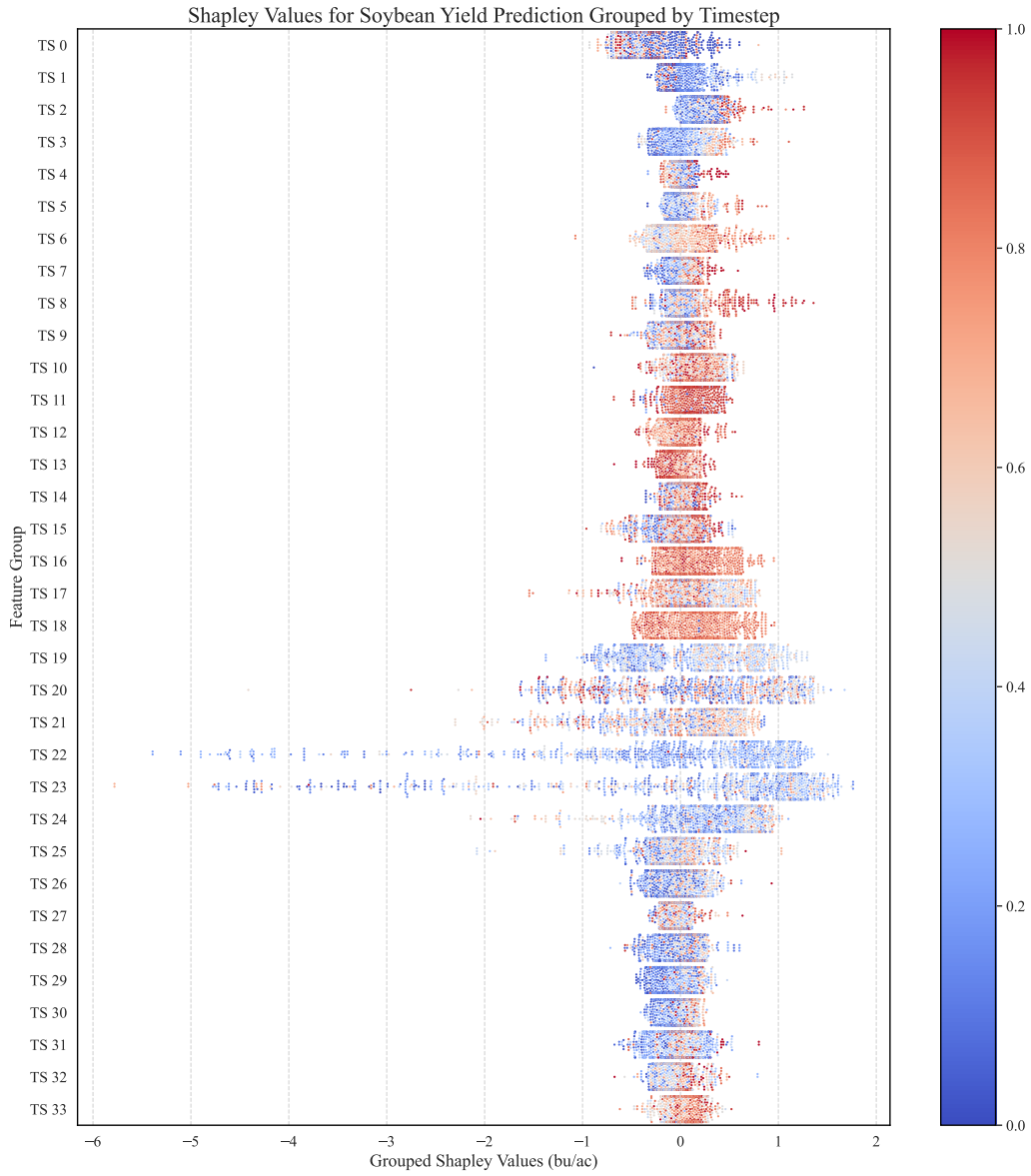


Figure 5.2: GSV feature importances for soybean yield prediction. Every dot is a local explanation of a soybean yield prediction in the U.S. Each plot represents all features captured at the specified time step (TS), running in 8 day intervals from the 49th day of the year. We see the biggest impact given by TS 22 and TS 23 during mid-reproductive growths. This shows the model decision making process to be in line with experts knowledge (Yang et al. 2023).

predefined coalition in each run would be the one that should be analyzed, and the other would be composed of all other features. As the sum of the two GSVs will be the same in both cases, the importance of the regarding predefined coalitions can be compared. Furthermore, our visualization works best if the value of the feature groups can be represented meaningfully by an aggregated value. Lastly, since fast calculation of the GSV is only enabled for tree structures, it remains an open problem how to handle explanations for models with high-dimensional input data that do not rely on tree structures. Next in this thesis, we will experiment on using the calculated feature importances to select features and create lightweight models for yield prediction.

5.3 Application-Driven Discussion on Shapley Values for Feature Selection

As we have successfully proven the worth of GSV for explaining models, we now want to discuss the question whether we can use them to select meaningful feature subsets to rebuild our yield prediction models. This section is dedicated to an application-driven discussion on the capabilities of Shapley values for feature selection. To do so, we will test three different options to select features to preserve model performance in well-known real-world data sets. As a baseline and representation of the state-of-the-art, we will use the internal Gini feature importances of random forests. We show how a greedy selection of features by Shapley value feature importances improves over the state-of-the-art and evaluate the impact of model averaging by discussing the results of the CFS algorithm as introduced in Section 4.2. First, we will give an in-depth explanation of a selected real-world example that highlights the differences between CFS and greedy Shapley value feature selection. Following this, we show the results of further experiments on different data sets, before we use the results of the feature selection process to replicate the experiments of Section 5.1 with a smaller feature space. Most of the content in this section has previously been published (Huber & Steinhage 2024d).

Experimental Design

To evaluate the three different feature selection methods and to build our base model M, we again use XGBoost to build a random forest. As explained above, XGBoost builds the tress sequentially, where each tree is designed to reduce the residual error of the already existing ensemble. For hyperparameter tuning, a Tree-structured Parzen Estimation (TPE) (Bergstra et al. 2013) is applied. This is a sequential model-based optimization approach. Models are constructed sequentially to approximate the performance of the model for a set of hyperparameters based on historical measurements. TPE estimates the underlying relations between a quality measure and the hyperparameters by exposing the underlying expression graph of how a performance metric is influenced by the hyperparameters. For this study, the Python implementation of TPE within the Optuna framework is used to tune the hyperparameter for each selection of subsets (Akiba et al. 2019) and ran for 25 iterations without any additional pruning involved to optimize accuracy in a 5-fold cross-validation of the training data.

The three different feature selection methods evaluated are implemented as follows.

First, we use the internal Gini feature importances of the random forests as a baseline and representation of the most used feature selection methods in practical applications. The feature selection is done by greedily selecting the highest scoring feature according to the internal feature importance from XGBoost as a baseline selection. To calculate the internal Gini feature importance for a feature, we evaluate all splits made with the help of the features. The Gini entropy is calculated according to the amount of training data that are divided by the split and, lastly, all added up. As a second approach, we will evaluate the feature selection achieved by greedily selecting the feature with the highest absolute Shapley value in each iteration. This method represents the core question of this section, whether Shapley values can be used for feature selection in such a fashion. Lastly, we have our novel approach of CFS that can be seen as a way of using the idea of Shapley values for feature selection, without suffering from model averaging, and thus can be used to evaluate the impact of model averaging on different data sets and raise our understanding of the problem. We performed our experiments by selecting a subset of every size with each of the three feature selection methods. We then optimize and train a new model on the subset of features to report the resulting accuracy on the testing data. With this procedure, good feature selections should result in high accuracy and low prediction error, giving us a ranking of the feature selection processes.

Explaining Model Averaging on a Real-World Problem

We start our experimental evaluation by taking a more in-depth look at a real-world problem to understand the conditions that cause model averaging to be a problem for Shapley value feature selection. For this purpose, we will use a data set not related to yield prediction, as we need a data set with less complex feature interactions that allows for a better investigation of the results. We use the bike rental data set from the UCI machine learning repository (Fanaee-T 2013). The data set is often investigated within the literature (Sathishkumar & Cho 2020) and has many advantages in explaining feature selection. Most importantly, we can understand the problem and the features involved. The target variable is the number of bikes rented in an hour through the Capital Bikeshare system in Washington DC, USA. The features we use are described in Table 5.3. We chose this data set to discuss model averaging, since its features have interesting relations. The feature year is important without considering any other feature, since the values for 2013 are generally higher than 2012. The mean for all data points in 2012 is 143 bikes rented, and for 2013 it is 234 bikes rented. The features of holidays, weekdays, and workdays also have an interesting relationship with each other. The weekday and holiday features can be used to reconstruct the workday feature, since a workday is a day not on the weekend, where no holiday occurs. Finally, we see a strong correlation between temperature and perceived temperature, with a Pearson correlation coefficient of 0.988.

For this data set, we selected subsets of all sizes from 1 to 11 with three different selection procedures. In the analysis, we focus especially on the difference between the Shapley value feature selection and CFS. The results are shown in the top left part of Figure 5.3. Differences occur for selections of sizes 2, 3 and 4. The Shapley value feature selection for size 2 consists of the features year and hour, while CFS selects temperature and hour. While the hour is the same in both selections, we are able to see a problem

with model averaging for the Shapley value feature selection. This is because the feature year is important when adding it to any subset of features, which is good for achieving a high Shapley value. However, the feature temperature is more important to estimate the number of bikes rented. The temperature variable is selected later by the Shapley value feature selection, since sets that already include the highly correlated perceived temperature lower the Shapley value feature attribution and delay the selection of an important feature. This shows how model averaging can be problematic.

For the feature selection of size 3 the Shapley value feature selection method selects the feature temperature next. The CFS algorithm adds the feature workday to the feature selection. Again, due to model averaging, the Shapley value feature selection does not select the best feature, as year again adds value to every possible subset, but the feature workday is not yet chosen due to the relations to the other features weekday and holiday. For subsets of size 4, both approaches add the feature rush hour to the selection, still inheriting the problems of the previous selection. From now on, both approaches add the same features to the selection. Furthermore, both approaches consistently beat the Gini feature importance in terms of RMSE.

Feature	Description
year	The year the bike rental took place. Either 2012 or 2013.
month	The month the bike rental took place. Values 1 to 12.
hour	The hour during the day the bike rental took place. Values 1 to 24.
holiday	Was the bike rented on a holiday. Values 0 or 1.
weekday	Day of the week the bike rental took place. Values 1 to 7.
workday	Was the bike rented on a regular working day? Values 0 or 1
weather situation	Integer evaluation of the weather condition. Values from 1 - good to 4 - bad.
temperature	Temperature when the bike rental took place. Normalized between 0 and 1.
perceived temperature	Perceived temperature when the bike rental took place. Normalized between 0 and 1.
humidity	Humidity when the bike rental took place. Normalized between 0 and 1.
windspeed	Windspeed when the bike rental took place. Normalized between 0 and 1.
rush hour	Was the bike rented during rush hour (7:00 to 9:00 or 17:00 to 19:00)? Values 0 or 1.

Table 5.3: In depth description of the features used to predict the amount of bikes rented within the hour. This detailed table is the basis for explaining the impact of model averaging on using Shapley values as a feature selection tool on a real-world example. Table previously published (Huber & Steinhage 2024d).

Further Experimental Results on Different Data Sets

We solidify our results by performing more experiments on two data sets taken from the well-known UCI machine learning repository and additionally our application of soybean yield prediction with an extended feature space. The experimental setup is the same as explained for the bike rental data set. The first additional data set we analyze is about the prediction of air quality (Vito 2016). The data set consists of 9358 instances consisting of 12 features that can be used to estimate the hourly CO concentration measured in mg/m^3 an Italian city. The second experiment carried out within our work is the prediction of house prices in Boston (Harrison & Rubinfeld 1978). Here, the data set consists of 505 data points from the year 1970, where 13 variables describe features of the house and the corresponding neighborhood. The target price is the price of the house in 1000 $. Lastly, we include the data set to predict soybean yield in the US. We focus on the prediction of the yields in 2022. For our analysis, the features are grouped according to the satellite sensors used for data acquisition, together with one group for all handcrafted features, resulting in 13 different groups that can be selected during the feature selection process. To calculate the Shapley values we use the extension of Shapley values for groups of features as it is introduced in this thesis and use a similar extension for our CFS algorithm.

We can use the results of the experiments to gain more insight about the model averaging problem and the connected strength to mitigate model averaging with CFS. For model averaging to become problematic in Shapley value feature selection, we need two conditions to be met. First, we need to find two or more features that are correlated or contain similar information. Second, we need a different feature that has fewer capabilities when used within a prediction model but is mostly unrelated to the rest of the data set. Due to the Shapley value feature importance being averaged over the marginal contribution to every feature subset, the Shapley value of the features with similar information is lowered, everytime they are added to a subset where the other features are already present. This finally results in the feature that is mostly unrelated to the rest of the data set but with lower prediction capabilities to be selected first. Taking this insight, we can analyze the results in the air prediction data set in the top right corner of Figure 5.3. The selection from Shapley value feature selection and CFS is mostly the same, with the first distinction being made for feature subsets of size 8. This is because the seven most important features for the prediction problem are all highly correlated with each other, with absolute Pearson correlation coefficients above 0.9. The first difference in selection occurs when the previously described conditions are met. The features relative humidity and temperature are highly correlated with a Pearson correlation coefficient of -0.77, while the feature absolute humidity is left to select with all correlations within the data set being lower. CFS is able to correctly attribute a higher prediction capabilities of relative humidity, while the correlation with the temperature hinders the Shapley value feature selection to do so. For soybean yield prediction, we see that CFS and Shapley value feature selection produce identical feature subsets. This can be explained by using groups of features in this example, where highly correlated features can only be selected as a whole group. In this scenario, it is very unlikely that the remaining feature groups contain similar information, and model averaging is not a real issue for this data set, resulting in CFS and Shapley value feature

	Average RMSE over all subset sizes (Runtime)		
Data set	Shapley value	CFS	Gini
Bike Rental	71.38 bikes (1.02 s)	66.79 bikes (21 m)	81.68 bikes
Air Quality	0.27 mg/m^3 (0.42 s)	0.26 mg/m^3 (7 m)	0.28 mg/m^3
Soybean Yield	4.76 bu/ac (3 s)	4.76 bu/ac (18 m)	5.29 bu/ac
Boston House Price	5.63 k$ (0.27 s)	5.55 k$ (11 m)	5.64 k$

Table 5.4: The averaged accuracy of the models trained on the feature subsets selected with the three different feature selection approaches, with the best values in each row highlighted in blue. The regarding units are explained in Section 5.3. We see a general tendency for the CFS algorithm to outperform the competition. However, the differences between the greedy Shapley value selection and the CFS algorithm stemming from the problem of model averaging within the Shapley value feature selection are rather small. A similar table is previously published (Huber & Steinhage 2024d).

selection being equally as potent. Lastly, we have the Boston house price prediction data set where again the first feature selections are the same. A change occurs for the fourth selected feature where CFS selects a feature describing the amount of nitrogen oxides in the air that is highly correlated with two other features, the amount of industry in the area and the average distance to next Boston employment centers, finally leading to Shapley value feature selection providing different results. Summarized, the experiments show how different correlations within the data set can hinder Shapley values to find the optimal selection of features based on features that are not part of the optimal selection of a given size.

Throughout all experiments, we observe similar behavior of the three different feature selection approaches. Looking at the averaged RMSE values for all possible subset sizes in Table 5.4, we see that the Shapley value feature selection approach and CFS are always very close together in terms of accuracy, with the CFS algorithm having a small advantage for every data set but for the soybean yield prediction, where both approaches produce identical feature subsets. On the same note, we see greedy feature selection according to the internal Gini feature importances, which performs significantly worse for all our data sets, when compared to the other approaches. The detailed illustration of the results in Figure 5.3 shows the same picture as the table. The green and orange lines, showing the results for CFS and Shapley value feature selection, respectively, are always very close together, with the blue line showing the results of the internal Gini feature importances mostly lying above the other two approaches.

Reduced Feature Space for Yield Prediction

In this subsection we will use the above results to reduce the feature space for the yield prediction scenario shown in Section 5.1. As indicated by the graph on the bottom left in Figure 5.3, we see no real benefit in including more than the four most important feature groups according to the different bands of remote sensing data in building our model for the prediction of soybean yield. In this case, the resulting feature selection is the same, whether we build it on CFS or on Shapley values directly. As can be seen in Figure 5.1, the most important groups are the (1) handcrafted features, (2) the near-infrared

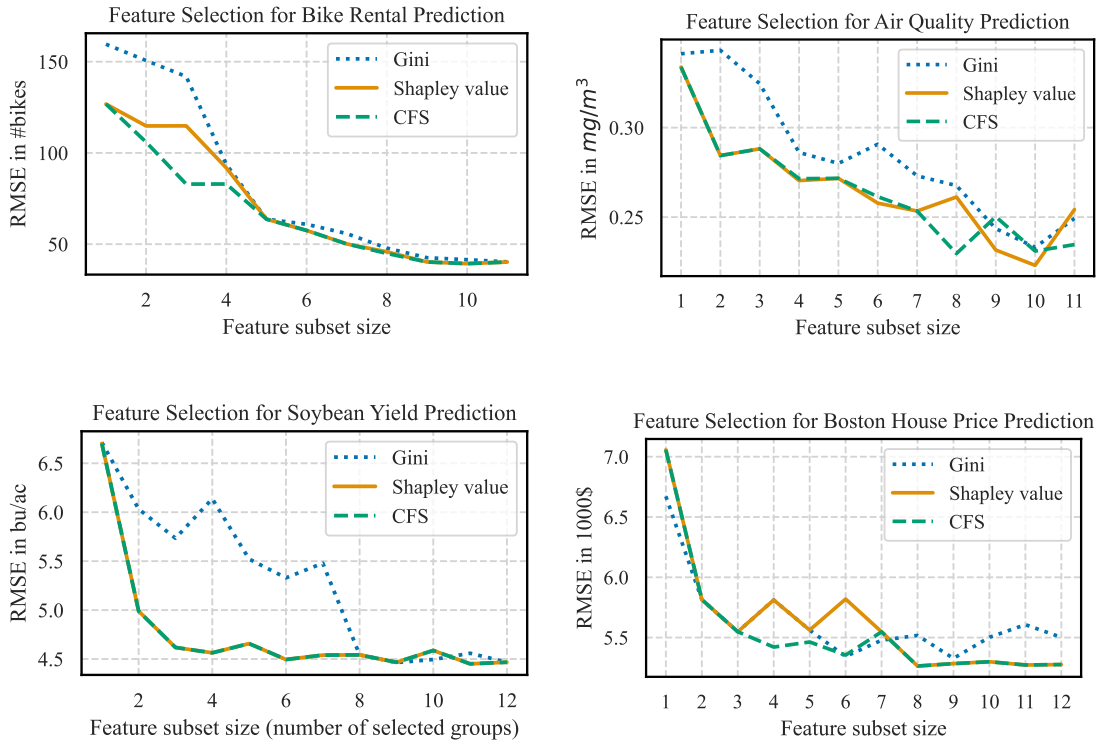


Figure 5.3: Experiments on selecting features with the three different approaches. The experiments indicate that both greedy feature selection according to Shapley values and the selection with CFS constantly outperform the often used greedy selection according to the internal Gini feature importances from random forests. This is indicated by the blue line, which mainly shows the largest prediction error for the different feature subset sizes. Figure previously published (Huber & Steinhage 2024d).

band with wavelengths 841 to 876 nm, (3) the infrared band with wavelengths 2105 to 2155 nm, and (4) the temperature during the day. This selection is in line with experts' knowledge concerning yield prediction and the results of building a novel model on the selected features are shown in Table 5.5. We see that the loss in accuracy is very small and for the years 2018 and 2019 the prediction on the feature subset is even slightly better than the prediction on the full data set in terms of RMSE.

Discussion

Given a machine learning problem with a data set and an existing model M, the subject of this section is the research question, whether Shapley values can be used as a tool to select a subset of features in a way that reduces the amount of input features and therefore the overhead in data acquisition, while the accuracy of the model M is conserved as much as possible. When approaching this question from a theoretical standpoint, we are able to formulate five possible pitfalls which could hinder a successful usage of Shapley values as a feature selection tool. Four of those are exposed by related literature to the

Year	RMSE Full	R2 Full	RMSE Reduced	R2 Reduced
2017	3.77	0.82	4.03	0.78
2018	4.51	0.76	4.39	0.76
2019	4.21	0.76	4.12	0.76
2020	4.22	0.80	4.29	0.80
2021	4.55	0.82	4.60	0.81
Avg	4.25	0.79	4.29	0.78

Table 5.5: RMSE in bu/ac and R^2 values of end-of-year soybean yield predictions for the full data set in comparison to the reduced data set. We selected the 4 most influential feature groups according to Shapley value feature attributions to rebuild the XGBoost model. The best result for every year is highlighted in blue.

topic and are focused on constructing counterexemplified examples for Shapley values as a feature selection tool by defining value functions that lead to the Shapley value feature selection to violate certain desired properties. The first contribution of our work is to define the four conditions C1 to C4 that need to be fulfilled by the value function, when Shapley values should be used for feature selection. We found that the conditional value function as defined by Algorithm 1 can satisfy all the conditions. The experiments show that the resulting definition of Shapley values is indeed suitable for feature selection and outperforms the often used internal Gini feature importances of random forests, when both are used within a greedy feature selection algorithm. Compared to other related applied work on the topic as presented in Section 2.3, this is in line, even though the Shapley value feature selection suffers from model averaging. The Shapley value feature importances are a better heuristic for the greedy feature selection, as the Gini importances only focus on the amount of training examples that are divided by a feature, but not directly on the impact on the target variable, when the feature is changed. This is the strength of Shapley values in machine learning and leads to a better quantification of the importance of a feature to the model and therefore to an importance score, well suited to select the most important feature to preserve the models performance on a subset.

However, we further analyzed that there is a problem with Shapley values for feature selection that cannot be solved through the choice of the right value function. Shapley values, by definition, suffer from the problem of model averaging when they should be applied for feature selection. This means that the Shapley value of an individual feature is influenced by features that are not in the desired final selection, and the greedy selection according to Shapley values will be suboptimal. We introduced a novel approach, CFS, to isolate the impact of unselected features and measure the impact of model averaging in several real-world problems. The experiments in general support the existence of model averaging as a problem, where CFS outperforms Shapley value feature selection on all selected data sets, as shown in Table 5.4. In-depth analysis of the different feature selections on the bike rental data set in Section 5.3, shows that the correlation between features and in particular the possibility of reconstructing the feature values from one another is a crucial factor when model averaging hinders Shapley value feature importances to indicate the optimal selection of features. This conclusion is also supported by the soybean data set, where similar features can only be selected as

groups and, therefore, the respective groups cannot be reconstructed from the remaining ones. That being said, we need to note that the CFS algorithm is computationally very heavy, when compared to Shapley values, at least for random forests, where Shapley values can be calculated in polynomial time. This hints into a research direction, where the use of CFS is further explored. CFS is formulated for random forests but the idea of evaluating the same value function as used in Shapley value calculation to minimize the error for feature selection can be transferred to other machine learning algorithms. For example, there exist Shapley value calculation procedures for deep learning applications. When exploring these options, the Shapley value calculation loses its advantage of fast calculations, as calculating Shapley values on arbitrary models remains NP-hard. Furthermore, restricting the number of subsets that should be considered for feature selection can further improve computational complexity and allow the evaluation of problems that include a larger feature space with CFS. For the problem at hand, the improvements in accuracy that we were able to find with our experiments indicate that the trade-off between computational complexity and gained accuracy when circumventing the problem of model averaging hints to use Shapley values for feature selection for applied problems. Finally, the best solution is to be picked individually by deciding on a trade-off between accuracy and runtime, between CFS and Shapley values for feature selection, while both approaches are to be preferred over using the internal Gini importances of random forests for feature selection. On the example of soybean yield prediction in the US, we see how we can build more efficient models using Shapley values of CFS for feature selection. We decrease the number of input features by 72% while only sacrificing 1% in performance, showing the capabilities of GSV for feature selection for yield prediction specifically.

5.4 Interpolating Gaps in LST Data

In this section, we first discuss a state-of-the-art statistical approach to fill gaps in LST data, followed by a comparative evaluation to show the capabilities of our approach. The results have previously been published (Huber et al. 2024b).

A Statistical Approach to Interpolating Remote Sensing Data

As a competitor and to evaluate the capabilities of our approach, we opted to use our own implementation of a state-of-the-art procedure for statistical interpolation of LST introduced by Metz et al. (2017). They present their approach to fully reconstructing LST data for central Europe at a resolution of 1 km in a two-step process, even including elevation and emissivity. The elevation helps to recognize temperature patterns induced at different height altitudes and can be used, regardless of how occluded the LST data are, since the elevation never changes. The same can be said for the emissivity. Although the emissivity can suffer from occlusion, like the sensors used to derive the LST, the emissivity is fairly constant and can be interpolated linearly over time.

The first step in the statistical reconstruction of LST data is the reconstruction in time. The original work of Metz et al. (2017) performs a local weighted regression with polynomial order 2, considering the five nearest neighbors in time, with gaps greater than seven

days not being interpolated. Extensive hyperparametertuning via Tree Parzen Estimation (Bergstra et al. 2013) and the Optuna framework (Akiba et al. 2019) to optimize on our validation data set showed that, for our particular use case, polynomials of order 5, considering the seven nearest neighbors, and interpolating any gaps up to a size of nine days, improve the results. This could be due to the fact that the weather in our study region is of high variance, compared to the entire region of central Europe as used in the study of Metz et al. (2017), where more moderate climatic conditions are prevalent. Values that cannot be interpolated by the first step are spatially interpolated using Thin Plate Spine (TPS) interpolation (Craven & Wahba 1978). Again hyperparametertuning revealed that a relatively strong smoothing coefficient of five helps to achieve the best results, and while the emissivity as a covariable of TPS proved itself helpful for interpolation, as suggested in the original work, the elevation was not useful to reduce the interpolation error. This could be explained with worse results, when the valid information was only accessible in the high- or low-laying parts of our interest region, as elevation as a covariable cannot provide additional information, without at least some data points being available for the different value ranges. For a fair comparison, we also added the up to 20 value captures from local ground weather stations to the LST images if they were missing, as they are computed according to Section 3.2.

Deep Learning Parameters

The experiments with our deep interpolation approach are performed on a NVIDIA Quadro RTX 6000. The batch size during training is set to 6. During training, the well-known Adam optimizer is used (Kingma & Ba 2014), with a learning rate of 4×10^{-5}. The learning rate decays by the factor 0.1 after 15 and again after 30 epochs. The network is trained for a total of 100 epochs to guarantee convergence. All parameters are selected on a validation data set, solely reserved for this purpose, consisting of 140 LST images with no occlusions of the ground-truth, and, therefore, all information available. The network input is normalized to the interval $[0, 1]$ using the minimum and maximum temperature values of the training data set.

Comparative Evaluation

To achieve a fair evaluation, we excluded 140 images with full data coverage as testing data during the entire modeling process. The test images are processed as training samples, explained in Section 4.4, resulting in partially covered images as input to the different approaches, but with the full ground-truth available. This process of masking pixel values according to the missing value pattern of a randomly selected training example removes an average of about 2500 pixels from the 4012 available pixels per image. The numerical evaluation of those experiments is shown in Table 5.6, which shows the capabilities of our deep interpolation approach. We improve the RMSE of the statistical approach by 44%, while being able to deliver a gapless reconstruction, even when the only valid pixels available are derived from the inferred data from the local weather stations.

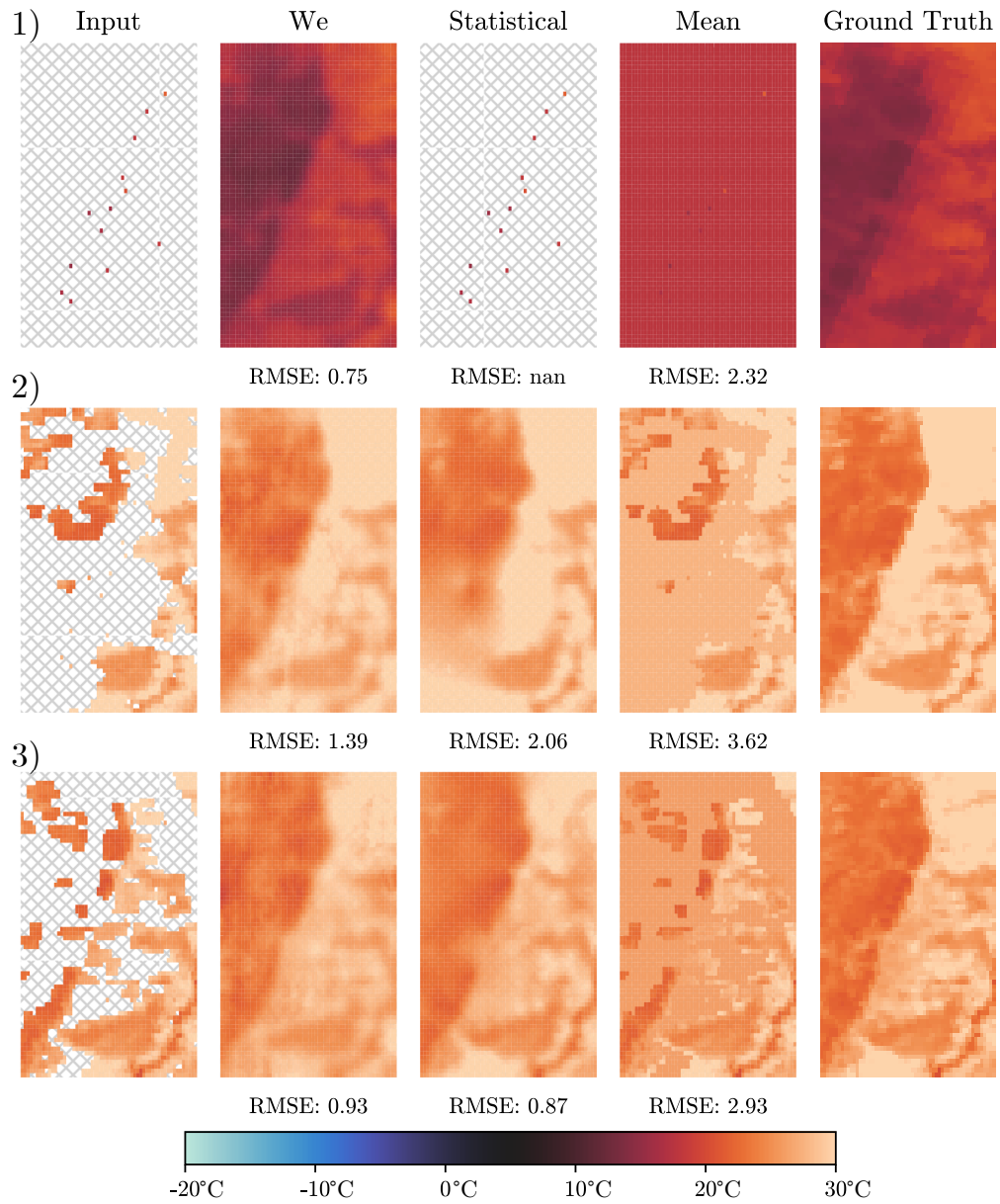


Figure 5.4: The reconstruction of three typical training images found in our data set taken on 29.09.2008, 09.05.2008, and 15.04.2008 respectively. In the first row, our research area is fully covered by clouds or cloud shadows, and no LST information is available. In the second row, more data are available, and in the last row, data are available, especially in the challenging area where the elevation changes drastically, allowing the statistical approach to do a great job of reconstructing the gaps. Figure previously published (Huber et al. 2024b).

Approach	RMSE	R2	Reconstruction
Mean Approximation	3.23 °C	0.81	100 %
Statistical Approach (Metz et al. 2017)	2.52 °C	0.91	52.06 %
Deep Interpolation (We)	1.67 °C	0.95	100 %

Table 5.6: Results for evaluating the accuracy via RMSE and R2 error metrics on 140 test instances for LST reconstruction. The 140 instances have full coverage of ground-truth data for the day. The value of pixels is masked according to the missing value pattern of a randomly selected training example, removing an average of about 2500 pixels from the 4012 available pixels per image. Table previously published (Huber et al. 2024b).

Discussion

When comparing our approach with state-of-the-art statistical methods, we see improvements in two main directions. First, the interpolation always covers 100% of the research area and second, the general precision is improved from 2.52 °C to 1.67 °C in terms of RMSE. The reason for the first improvement is that the statistical interpolation problem can be ill-posed if only a few data points are available. The first row example in Figure 5.4 shows a testing instance, where the selected cloud occlusion covers the entire image. Therefore, only pixels with valid information are reconstructed from ground-site weather stations. We see that our deep interpolation approach is capable of reconstructing the whole research area with all its environmental constraints, while the statistical approach is not capable of interpolating. The reason for the second improvement direction can be explained by looking at the second and third examples in Figure 5.4. The test instance shown in the second row shows a reconstruction error of 2.06 °C for the statistical approach and 1.39 °C for our deep interpolation. Looking at the reconstructed images, we see that the statistical approach is not capable of reconstructing the lower left corner of the image correctly. The ground-truth image shows the steep temperature gradient typical for our research area, which runs from the lower left corner to the upper right corner of the area. The statistical approach cannot reproduce this peculiarity when the information around this gradient is missing, as is the case in this example. For the third row example of Figure 5.4 we observe two very good reconstructions, with error values of 0.93 °C and 0.87 °C for our deep interpolation and the statistical approach, respectively. The high accuracy of the statistical approach can be explained by the existing data in the input image. Although the general number of data points is not too high, the available information is located mostly in the area where the reconstruction is the most challenging, allowing the statistical approach to slightly outperform our deep interpolation. In general, our approach is capable of learning the environmental constraints of the research area and reproducing them, even when crucial information is missing, giving an edge over statistical methods that see each instance as an isolated problem, without being able to induce knowledge from other instances from the research area. One way to help statistical approaches with this problem is to provide additional information in the form of covariables during the interpolation process. The covariables investigated within this study are elevation maps and emissivity. Elevation maps are not included in the results shown in this investigation, as our experiments indicated that including them reduces the accuracy of the reconstruction. However, the emissivity, which is fairly constant and therefore can be easily interpolated, even throughout cloud occlusion, helped to improve

the results and added some form of environmental context to the statistical model.

As explained earlier in the manuscript, the process of obtaining LST information from satellite records is complex. So, naturally, the question arises whether further processing of the data records with deep learning alters or impacts the data inherently. The first influence on the data comes from the integration of air temperature when interpolating LST. The difference between air temperature and surface temperature varies with respect to different surfaces, which poses a potential source of uncertainty within the interpolation. We minimize the potential for errors by training an individual model for every ground site weather station and, therefore, allowing the models to account for different surfaces. As shown in Table 4.1, the results for the different stations are very similar. The question whether the deep learning pipeline itself alters the data can only be answered by looking at the performance on the test data. The very high correlation between predicted temperature and real temperature indicated by an R2 score of 0.95 is a clear indication that our approach is true to the data at hand. However, there is a common problem when evaluating LST interpolation, which comes from the fact that the data used for training and evaluating the approaches have the inherent bias of being captured when no clouds are present. This so-called cloud-free assumption is necessary, as it is not possible to achieve ground-truth LST measurements for already occluded data points on a large scale. For our region of interest, the weather station at the ground site also does not provide direct information on LST that could be used to evaluate LST reconstruction without the cloud-free assumption, at least for some selected pixels.

Although the high specialization of our approach to the research area improves accuracy, it is harder to apply when considering a different area. For a new area, the statistical approach can be used out of the box and just needs access to the very instance that needs to be reconstructed. For our deep interpolation, it is necessary to find a database of training examples that can be used to train our deep interpolation before the reconstruction can be applied. However, building such a database is fairly accessible, since our proposed learning process can be executed exclusively on occluded ground-truth data, as explained in Section 4.4. To fully unlock the potential of deep interpolation in a new research area, the availability of local ground-site weather stations is also necessary. Although the approach would work in most cases without them, we would not be able to provide a 100% coverage of the reconstruction. When changing the research area, the amount of improvement of our deep interpolation approach over the statistical approach could also change, as the results shown in Figure 5.4 indicate the difficult inhomogeneous areas of the region, where our approach outperforms the competition.

5.5 Transferring Knowledge for Soybean Yield Prediction

To analyze the capabilities of transfer learning for yield prediction, we analyze the accuracy of yield prediction models built with different configurations of regularization techniques and the use of a Gaussian process. First, we examine the models for the end-of-year-prediction, including the full crop growth cycle, before conducting experiments on the performance of an in-year prediction. This section is based on our previous research (Huber et al. 2024a).

End-of-the-Year prediction

We compare the application of different transfer learning methods for the end-of-season prediction covering the entire crop growth cycle, with the results shown in Table 5.7. To account for the influence of random parameters, all values are determined over four runs each. For the Argentine models, the application of the Gaussian process improves the accuracy in all cases by up to 10% in terms of RMSE and by up to 25% in terms of R^2. Transfer learning approaches with regularization improve the results further. First, we examined the application of an initialization with the US model for all six layers with a freezing of the parameters of the first 4 layers. We see initial improvements compared to a model without any transfer learning applied that diminish after the concatenation of the Gaussian process. Incorporating the regularization techniques explained in Section 4.5 stabilizes the training and subsequently gives the best results. The overall best configuration is an initialization with the weights of the US model, a freezing of the first four layers, a fine tuning with using the L^2-SP regularization and lastly the application of the Gaussian process. Using BSS as an additional regularization slightly decreases the average accuracy but stabilizes the training process, as can be seen in Figure 5.5.

Figure 5.5 shows the distribution of the RMSE in bu/ac for each method. The first two boxplots show the basic drop in RMSE when the Gaussian process is applied. The US initialization with freezing subsequently causes a wide dispersion of the error values, indicating that transfer learning in general can be helpful but must be guided by regularization. Although BSS places the center of these scattered error values at a low level, L^2-SP causes the values to be centered at a lower value, indicating a lower average error. The simultaneous use of BSS and L^2-SP further reduces the scattering of error values, indicating that predictions provide greater reliability.

Approach	RMSE	R^2	RMSE + GP	R^2 + GP
USA: CNN	5.94	0.583	6.81	0.439
Argentina without transfer	7.47	0.442	6.76	0.554
Argentina + freezing	6.94	0.526	6.85	0.547
Argentina + freezing and L^2-SP	6.80	0.550	6.25	0.618
Argentina + freezing, L^2 and BSS	7.05	0.516	6.43	0.593
Argentina + freezing, L^2-SP and BSS	7.07	0.511	6.31	0.608

Table 5.7: Average RMSE in bu/ac and R^2 as fraction of 1, of different model configurations for the end-of-year-prediction. The best result is highlighted in blue showing the advancements of regularized transfer learning. Table previously published (Huber et al. 2024a).

In-Year prediction

As described in Figure 4.11 we examine a second shorter time frame of available information for our prediction models. A model capable of inferring the estimated yield way before harvest has a very high value for crop management. Table 5.8 shows a summary of the prediction results in the short period. Here, satellite images no longer include

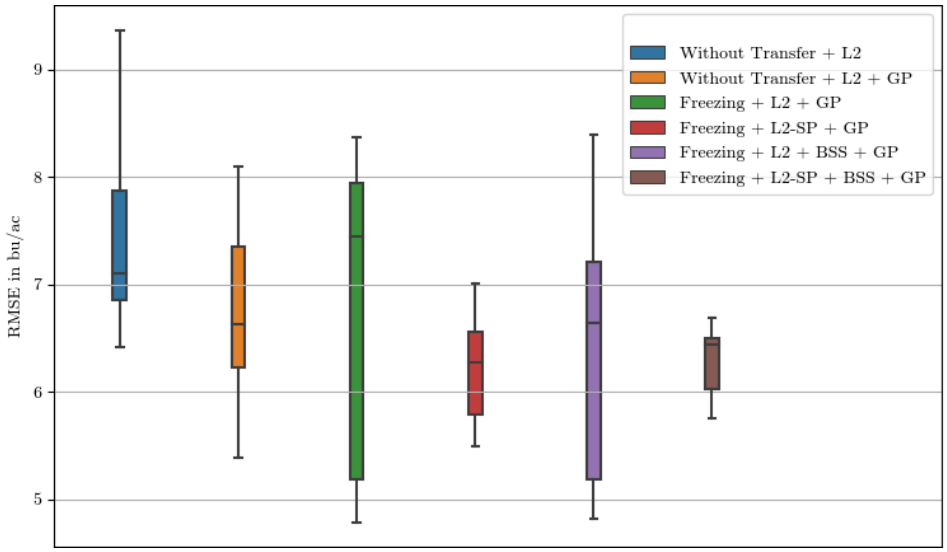


Figure 5.5: Distributions of the averaged district errors of all runs in all test years as boxplots. In this case, freezing implies the initialization of the 6 convolutional layers by the US model without further fine tuning of the first four layers. Involving BSS into the pipeline helps to decrease the variance in error for different models. Table previously published (Huber et al. 2024a). Figure previously published (Huber et al. 2024a).

Approach	RMSE	R^2	RMSE + GP	R^2 + GP
USA: CNN	7.12	0.394	7.00	0.414
Argentina without transfer	9.36	0.140	8.37	0.314
Argentina + freezing, L^2-SP and BSS	8.20	0.349	6.92	0.537

Table 5.8: Average RMSE in bu/ac and R^2 as fraction of 1, of different model configurations for the in-year prediction. Again showing the capabilities of regularized transfer learning. Satellite image coverage ends before the start of the harvests. Table previously published (Huber et al. 2024a).

the harvest and end before the first harvest begins. Training, evaluation, and testing are performed exactly as in the long period. We used only the first 14 satellite images, instead of 34 as done previously. This is reflected in a reduced performance compared to the long period. At the same time, it can be seen that transfer learning methods, especially with simultaneous application of the Gaussian process, result in an even greater increase in performance.

Discussion

While XGBoost was superior in our initial experiments for the prediction of soybean yield in the US, we chose to explore CNNs as the second-place method for our transfer learning experiments. This is due to the proven capabilities of CNNs for transferring low-level features learned in one domain to a new domain, as it is used in transfer learning (Gupta et al. 2022). Transfer learning approaches based on random forests

are often only feasible when both domains are very close to each other, as it is not possible to distinguish between low-level features that can be transferred and high-level features that are domain-specific. Furthermore, we decided to add a Gaussian process to the predictions. The addition of the Gaussian process for predicting US yields is beneficial for the short prediction period, while decreasing the accuracy for the long prediction period. For the Argentine yield prediction without transfer learning, the Gaussian process improves the results by ca. 10% in terms of RMSE for the long period and the short period. Similar results are discovered by Kaneko et al. (2019) where a Gaussian process improves prediction results based on small data sets to a level similar to that that can be observed when training with more data. The same effects are shown when examining the first step of transfer learning for Argentina, where the CNN weights are initialized with the US model's weights, and the first four layers are frozen. Although the improvement without a Gaussian process is about 7% in terms of RMSE, the results are worse when the Gaussian process is included. Getting worse results with a machine learning model after application of knowledge transfer, as we have in the case of our model when the Gaussian process is included, can be described as negative transfer and is well anticipated in the literature (Pan & Yang 2009). As yield prediction is a research area that suffers from data scarcity, the selection of a source domain for pre-training is prone to a known tradeoff. On the one hand, more data for pre-training improves the results, while on the other hand, less similarity between the domains endangers the knowledge transfer. Our experimental setup tends to emphasize the amount of training data for pre-training over the similarity of the domains, as in real-world yield prediction applications it is often not possible to produce more training data close to the target domain. Despite this choice, the first transfer experiments indicate that features extracted from remote sensing data can be transferred similarly to those obtained in many computer vision tasks. This claim is supported by the fact that we were able to freeze the first four layers in our CNN that commonly condense high-level features from the data and improve the model accuracy without the Gaussian process.

The negative transfer that is prevalent when we evaluate the models with the Gaussian process can be addressed by regularization techniques. Our results support the claim that regularization techniques designed for commonly used image features have similar effects on remote sensing data presented as histograms. The L^2-SP regularization together with the Gaussian process gives the best results in terms of average RMSE and R^2, removing the negative transfer that occurred without regularization. As indicated by Chen et al. (2019) the inclusion of BSS gives us a small decrease in average performance by 2.8% in terms of RMSE, but stabilizes knowledge transfer. As can be seen in Figure 5.5 the worst error values are closer to the average RMSE than in any other constellation, which makes us recommend the combination of layer freezing, L^2-SP and BSS for transfer learning tasks, including remote sensing data represented by histograms. The same constellation also works well when considering the short prediction period (Table 5.8).

Within the context of the wider literature, the first parallel between our work and related work also considering yield prediction with remote sensing data is the use of MODIS satellite data as a primary data source, as is done, for example, by You et al. (2017), Wang et al. (2018) and Khaki et al. (2021). Although all of those works report good results, it is worth mentioning that alternative data sources exist. Fernandez-Beltran et al. (2021) use the Sentinel-2 satellite to also achieve state-of-the-art results,

exploiting the fact that the images are available at a higher resolution. Furthermore, the literature is beginning to investigate the use of non-fixed time steps during histogram creation as input for machine learning procedures (Desloires et al. 2023), which may be useful to increase alignment between the two domains used for transfer learning in the future. When considering the literature for yield prediction in general, our results improve on state-of-the-art performance, as we can deduce from our comparison to successfully deployed deep learning architectures (You et al. 2017) trained and evaluated on our data. This increase in performance comes from enabling the US, as the biggest repository for ground-truth yield data, as the source domain for transfer learning, even for a country on a different hemisphere. This builds on the work of Wang et al. (2018), where transfer learning is first used to improve yield prediction through transfer learning, but the two countries used, i.e., Argentina as source and Brazil as target, are much closer related than in our case.

However, we mention that the improvements come at the price of many additional hyperparameters that have to be tuned. Tuning hyperparameters in a deep learning context is always difficult, since the impact of a hyperparameter can mostly only be observed after a significant amount of computations. This makes it so that research often turns to empirical values or educated guesses. Regularization and transfer learning include an additional six important hyperparameters to adjust: L^2-SP, BSS, the number of frozen layers, and the initialization of the non-frozen layers. In our experiments, the number of 4 frozen layers indicates that many features learned from remote sensing data in our target domain can be directly transferred to the new task. The high transferability is also indicated by the advantageous initialization with the source weights for the non-frozen layers. The L^2-SP hyperparameter α with a value of 0.23 quantifies the punishment for altering the weights of the source models. This value being relatively low means that the non-frozen layers must be able to be highly adjustable to the new task, hinting that the yield related patterns utilizing the frozen features of the first layers are quite different for both our yield prediction tasks. The high L^2-SP hyperparameter β is the standard L^2 punishment for high weights suppressing overfitting. The BSS hyperparameters η and k and their respective values 0.07 and 1 indicate that the strength of the regularization is relatively low and the smallest singular value is penalized. For all these hyperparameters, small adjustments can alter the models' performance, increasing the risk of a bad model due to careless handling of the hyperparameters compared to a simpler model without transfer learning.

6 Application of Explainable Yield Prediction in KI-iREPro

In this chapter, we will revisit the KI-iREPro project for grapevine yield prediction that motivated our contributions to the field of yield prediction with its practical example. First, we will look at the commercial side of grapevine yield prediction, where we make predictions for commercial vineyards at the plot level, with possible benefits in terms of logistic planning and financial security for winemakers. After this, we will look at plant-level predictions that will be especially helpful for the early identification of experimental breeding lines that can meet the yield thresholds to become commercially viable.

6.1 Grapevine Yield Prediction for Commercial Plots

In this section, we will analyze our experiments and insights on grapevine yield prediction within the KI-iREPro project using the data described in Section 3.2, giving a direct use case for our developed yield prediction pipeline and the interpolation of remote sensing LST data.

Experimental Design

We will analyze our experiments and insights on grapevine yield prediction within the KI-iREPro project. Using XGBoost as the primary prediction tool, as it has shown its capabilities for other yield prediction scenarios, our aim is to evaluate the capabilities of the approach for grapevine yield prediction. For this reason, we test not only the XGBoost-based approach but also a simple feedforward network, as well as a baseline average predictor. The feedforward network consists of six linear layers, each with batch normalization and a ReLU, with each layer having half the size of the previous, until the singular prediction is given. The data is used as described in Section 3.2. Our prediction data are most importantly filled LST records that model climatic conditions throughout the year. The data are aggregated in 10-day intervals to match our ground-truth yield data. Additional information includes soil, elevation, and historical yields, as well as an identification of the respective winemaker who harvested the parcel and some phenological information such as the variety and age of the plants. We select two different testing scenarios, one in which we exclude the data from a full year of harvests, where only the data captured before the testing year can be used to train the model. This is the more realistic scenario, as in real-life applications, we have no information about the yields of the ongoing year at the time of prediction. The other train-test split is motivated by the literature and poses an easier problem. For example, Sirsat et al. (2019) make predictions for a random 20% split of the data. This might not be the most helpful in the application, but the results can still help to decide on

a direction for further studies, finally showing a direction to achieve great results for the yearly predictions in the future. Later in this chapter, we will elaborate on how to improve on data acquisition in viticulture, to allow for more accurate predictions in the future. For our XGBoost model, we will tune again using Tree Parzen Estimation (TPE) (Bergstra et al. 2013) inside the Python Optuna framework (Akiba et al. 2019) for 30 iterations without further pruning. To evaluate the parameters during tuning, we use a five-fold cross-validation of the training data. This showed the best results, even when optimizing for the yearly prediction, as the data show great intervariability between the years, making a tuning procedure, where a whole year of data is used to validate hyperparameters unfeasible.

Results, Discussion and Use in Application

The results of our two experiments are shown in Table 6.1. Looking first at the random split prediction, we see an improvement of 0.06 in terms of RMSE and 0.206 in terms of R2. Although the improvement in RMSE is rather small, the gain in R2 is significant, indicating that our model starts to learn about the underlying relations of the yield data and is capable of explaining some of the variation for the target data, as could be shown by other grapevine prediction approaches with random forests (Sirsat et al. 2019). We also learn that for any prediction scenario, the deep learning approach in the form of a simple feedforward network cannot learn about the prediction of grapevine yield from our data. This might be due to the low amount of training data, compared to most applications involving deep learning, and the tendency of simpler random forest-based models such as XGBoost to better handle a low amount of training data. In all our deep learning experiments, the RMSE is high, and the negative R2 score indicates that the data are not well understood by the model. Looking at the second prediction, the yearly testing split, we again notice that the XGBoost-based approach is slightly better than the mean prediction in terms of RMSE and improves the average R2 score by about 0.059. Although these results are not ground braking, they show that we can learn about grapevine yield prediction with XGBoost and our data pipeline. The question remains as to why the grapevine yield prediction presented here is so much more difficult than the soybean yield prediction in Section 5.1. One reason can be found in the nature of the two crops investigated. While soybean seedlings are planted every year, allowing us to monitor the entire cycle of crop development, grapevines are planted years before harvest occurs, making it impossible to model their entire development. Furthermore, while soybean yield is maximized for quantity, making it irrelevant to try to model the processes applied to the plants, as they can be assumed to be fairly constant, grapevines are maximized with regard to quality. Without detailed information on the processing steps taken, for example, cutting whole grape clusters, reducing the yield by up to 40% (Palliotti & Cartechini 1998), there will always be fluctuations in yield that cannot be explained by our models. Lastly, notice that the quantity of our data is smaller and the aggregation by winemaker that we needed to implement to account for logistical circumstances in viticulture practice, make it even harder to distinguish the yield determining factors.

However, our efforts in grapevine yield prediction can serve as a hint to the capabilities of machine learning in viticulture. With this in mind, we developed a prototype for

commercial plot-level yield prediction that can be used as a starting point for future projects. The planning has already started for a novel project where yield data in viticulture will be captured on plot level precisely, without any yield-changing manual practices being applied to the grapes, to remove two of the biggest challenges with our current data set.

Test Split	XGBoost		Deep Learning		Mean predictor	
	RMSE	R2	RMSE	R2	RMSE	R2
2020	0.025	0.124	0.186	-2.006	0.028	0.027
2021	0.031	0.029	0.306	-0.388	0.031	0.005
2022	0.027	-0.007	0.863	-0.049	0.028	-0.041
Average	0.028	0.049	0.452	-0.814	0.029	-0.010
Random Split	0.023	0.210	1.235	-2.182	0.029	0.004

Table 6.1: Prediction results for the grapevine yield prediction in the KI-iREPRo project. The data were normalized to lay between 0 and 1. The best result is highlighted in blue. The experiments are two-fold: First, a yearly prediction close to the real-world use case and second, the testing data consisting a random 20% split of the data. The second experiment is easier for the machine learning models to solve but can be used to evaluate our experiments in the wider scope of the literature.

6.2 Grapevine Yield Prediction on Plant Level

Another use case for yield prediction in viticulture is given by prediction at the plant level that has direct access to the phenological traits of the plants. Here, we will predict the yield for a variety of experimental breeding lines.

Experimental Design

Following the results of the experiments in the thesis so far, we will use XGBoost as an algorithm suitable to build a regression model to predict yields with limited amounts of training data. Furthermore, we want to retrain the model every year with the new data to increase accuracy and make the model adaptable to deal with an ever-changing environment, especially in times of climate change. Therefore, the selected machine learning algorithm should be quickly re-trained. To obtain a proof-of-concept for the prediction of grapevine yield at the plant level, we trained both a random forest for regression through XGBoost and a linear regression on the same feature set. Following the experiments as previously published (Huber et al. 2024c), we evaluated two main metrics, the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE). To emulate real-world conditions in our experiments, even with the small data set, we created 11 testing scenarios, one for each grape variety within our data. Hyperparameters are tuned by optimizing the RMSE to achieve the best performance in a 5-fold cross-validation on the remaining training data. The average RMSE for the eleven varieties is 0.68 kg for XGBoost and 0.76 kg for linear regression, showing an improvement of circa 11%. Regarding MAE, we measure an average of 0.55 kg for XGBoost and 0.63 kg for linear regression, showing an improvement of approximately 13%. The concept provided to use XGBoost for grapevine yield prediction shows its capabilities even in a

small data set. A complete breakdown of the results of the individual varieties is shown in Table 6.2. The improvements of XGBoost over the baseline approach are predicted to increase with an increase in training data, since the more powerful model XGBoost will be able to learn more complex relations that determine the yearly yields. Our test scenario is close to the real-world use case, with the grape varieties tested missing from the training data. Therefore, our reported results hint towards XGBoost as the best choice to solve automated grapevine yield prediction in the future.

Results, Discussion and Use in Application

The results of these experiments must be taken into the context of limited input data and a difficult data set, which contains varieties that are not fully explored with respect to their possible yield quantities and are prone to higher variations. For example, De La Fuente et al. (2015) report slightly better results when predicting well-established varieties by statistical analysis with an RMSE of down to 0.46 kg. However, in general, we already see a positive correlation between the predicted and actual yields. The higher error comes from the tendency of our model to not predict yields higher than 2.5 kg, as those are underrepresented during model training. That being said, the expected maximum performance for machine learning-driven yield prediction is higher than that for statistical approaches, since they are able to model even the most complex relations in the data, when enough training data points are available. Furthermore, machine learning models are adaptive to changing conditions with regard to grapevines. In times of climate change, it is possible that statistical models need to be calibrated by experts, while a machine learning model can simply include new samples in the training data to adapt to the new situation. However, our model has already been applied at the Julius Kühn-Institut (JKI) in the context of breeding new varieties and early identification of grapevine breeds that can meet the requirements to become commercially viable. With future application in mind, we expect our models to become more precise as more training data becomes available in the years to come.

	Variety	Do	PN	PB	Ri	BL1	BL2	BL3	BL4	BL5	BL6	BL7	AVG
XGB	RMSE	1.17	0.47	0.80	0.63	1.03	0.52	0.50	0.68	0.68	0.46	0.54	0.68
	MAE	1.02	0.38	0.58	0.44	0.86	0.43	0.39	0.57	0.62	0.34	0.44	0.55
Lin. Reg.	RMSE	1.19	0.42	1.25	0.63	1.03	0.58	0.47	0.63	0.91	0.70	0.54	0.76
	MAE	1.06	0.33	0.99	0.44	0.88	0.51	0.34	0.54	0.86	0.58	0.44	0.63

Table 6.2: Experimental results for grapevine plant-level yield prediction on data captured via manual plant appraisal. The input data can be found in Table 3.2 and the varieties are explained in Table 3.1. The presented table shows the RMSE and MAE error value in kg when different varieties are used as test data comparing XGBoost and Linear Regression for yield prediction. The best result for each column and each metric is highlighted blue. Table previously published by Huber et al. (2024c).

7 Conclusions

To conclude this thesis, we want to revisit each of the challenges for yield prediction as explained in the introduction. For each of the challenges, we will quickly recap our solutions, the state of the related work before our contributions, ideas for further improvements, and the importance besides yield prediction.

a) Training on small data sets

To approach the challenge of training on small data sets, we explored XGBoost as a solution. We provide a processing pipeline to efficiently process remote sensing images to feature vectors for machine learning by modeling the underlying distributions of the data. By conducting experiments on one of the largest data sets available for yield prediction, soybean yield prediction in the US, we see that the derived feature-based representation of the remote sensing data allows for the successful application of XGBoost for yield prediction. Demonstrating state-of-the-art performance, when compared to deep learning-based yield prediction approaches, while offering faster runtimes and better options for explainability.

Before our work, deep learning approaches dominated the field (You et al. 2017, Sun et al. 2019) using a histogramization approach for remote sensing data. Those works did not account for the limited amount of training data available in yield prediction scenarios. Unlike this, our idea of extracting features from remote sensing data and using a machine learning algorithm known for handling small amounts of data is therefore able to improve the accuracy. However, deep learning is an evolving field. We have already tested novel developments in the form of convolutional transformer networks (Inderka et al. 2024) and found that they cannot yet improve over other deep learning approaches. This is again due to limited access to training data.

In the future, it must be acknowledged that the available data sets for yield prediction are growing yearly, and deep learning could catch up in performance or even overtake the XGBoost approach in terms of accuracy. One of the main points of interest will therefore be monitoring the accuracies of the already developed methods over the next couple of years. Furthermore, it will be interesting to investigate the exchange of the fixed eight-day composite of remote sensing data with flexible window sizes to improve the similarity of the data between different years. Some early experiments showed that this approach is very promising for further improving yield prediction with machine learning.

The exclusive use of remote sensing data during our experiments results in great reusability of our results. Remote sensing data are available not only free of charge, but also as historical records. As long as historical ground-truth yield data are available, our pipeline can be applied to any yield prediction scenario by matching the yield data with fitting remote sensing data. Looking at the impact of our results in addition to the field

of yield prediction, we see that remote sensing data are the input to several machine learning problems such as monitoring changing climate (Schneider & Hook 2010) or analyzing the environmental impacts of land cover (Gohain et al. 2021). Our insights on processing remote sensing data and the comparative evaluation revealing that for small data sets gradient boosting methods outperform deep learning can finally be valuable for a variety of other problems.

b) Need for explanations

With the establishment of XGBoost as an approach to yield prediction, which results in a random forest for the final regression model, we are able to concentrate our efforts on explainability in this direction. We define Grouped Shapley Values (GSV) to exploit natural groupings of features from yield prediction scenarios to improve the usability of Shapley values for explanations. For this, we give a polynomial algorithm to calculate the GSV for random forests, where instead of iterating over all subsets of players when calculating the Shapley value, we just iterate over every path of the trees. To achieve a global understanding of the model, we leverage the visualization of many local explanations. This is done by introducing swarm plots that not only show the GSV but also give a colored indication of the feature's aggregated values, giving further information to analyze. In addition, we provide proof that calculating the GSV is not the same as summing up individual Shapley values.

In previous efforts within the literature, there were yield prediction models that were explained by grouping features (You et al. 2017, Sirsat et al. 2019). However, obtaining those explanations was mostly done by retraining the whole modeling process on different feature subsets, which is very time-consuming. Furthermore, individual explanations are not suitable for yield prediction due to the high feature count and the abstract nature of the individual features. Therefore, they are rarely used in related work, making a straightforward application of Shapley values for yield prediction impossible. A polynomial calculation for the classic Shapley value for random forests was presented before (Lundberg et al. 2020) and served as a starting point for our GSV calculation. The idea of grouped Shapley values in machine learning was first explored by Jullum et al. (2021) without considering a fast calculation, making the use of summed individual Shapley values the most used idea for aggregated Shapley values (Redelmeier et al. 2020, Aas et al. 2020).

For future work, it is a promising idea to extend Shapley value feature attributions to provide instructions to take actions guiding the model towards a desired output. This will be especially useful when we can include factors like fertilization or watering in a yield prediction model, as they are features where the agent can directly change the input values. Additionally, as for now we focus on explaining the model behavior as a whole, we can work towards exploiting individual explanations to enable understanding of singular data points, as it will be very interesting for end users that use prediction models to make predictions regarding personal data points.

In the context of yield prediction, we can use GSV not only to learn from the predictions, for example, about conditions that favor high yields but also to raise trust in our models. This is achieved by showing that the important features are in line with the features that domain experts would consider for yield prediction. For soybean yield

prediction, we found that historical tendencies of specific counties to produce high or low yields are the most important, directly followed by infrared surface reflectance data, which are known to correlate with plant health. Within the KI-iREPro project, the GSV was a valuable tool to build a common language between viticulture experts and computer scientists and are integrated into the application now in use at the JKI. Now, domain experts can use the feature importances to analyze models themselves and give feedback on how a model can be improved. Looking further than yield prediction, explainable artificial intelligence (XAI) is an integral topic in machine learning research at the moment. Grouped Shapley values can be applied for any kind of machine learning problem where natural groupings occur. This includes most prominently all kinds of time series problem in which the same observations are repeated multiple times.

c) Selecting important features

When obtaining feature importances with GSV, the natural question arises whether we can use this information to select meaningful features for an optimized model. To help answer the question, we make two methodical contributions to the field. First, we define four necessary conditions that must be satisfied to make the Shapley values suitable for feature selection. Second, we expose some lost potential for Shapley value feature selection due to the nature of Shapley values as a model averaging procedure, where features that should not appear in the final selection can alter the selection process. We overcome those issues by defining a new feature selection method, Conditional Feature Selection (CFS). We use CFS as an exhaustive feature selection counterpart to Shapley values and can subsequently evaluate the impact of the model averaging problem.

Shapley values for feature selection are a controversial topic within the literature, with many existing works discussing different shortcomings of poorly defined Shapley values (Kumar et al. 2020, Huang & Marques-Silva 2023, Sundararajan & Najmi 2020, Fryer et al. 2021). While those works focus on exposing weaknesses of the Shapley value, we extend on their work by proposing the 4 necessary conditions for successful Shapley value feature selection. There were other promising applications of Shapley values in feature selection (Fang et al. 2022, Zacharias et al. 2022). However, the problem of model averaging was never isolated to be evaluated, as we did by introducing CFS. In addition to Shapley values, in practice often simple feature selection methods, as, for example, based on the internal Gini importance, are used. Although the procedure is rather simple and therefore fast to calculate and robust, it fails to acknowledge the magnitude of change in the regression result when a feature is used to make a decision within a tree. Shapley values and CFS do account for this.

When comparing the CFS and Shapley values, we see that the advantage of Shapley values is derived from faster calculation times. With this advantage only being present for random forests, it allows for further exploration of the idea of CFS when extending to other machine learning methods. Based on the definition of SHAP (Lundberg & Lee 2017), sampling is used to evaluate the value function during the calculation of Shapley values for arbitrary machine learning models. Extending CFS by this idea would remove the computational advantage of Shapley values and might make CFS the strictly preferred feature selection method. Additionally, proposing run-time constraints for CFS could further improve the run-time of the approach, as certain subsets might

not need to be evaluated according to, for example, size constraints.

In the context of yield prediction, our results allow us to build more efficient models with only a minimal decrease in prediction accuracy. An even greater benefit for feature selection can be found for the prediction of grapevine yield, where data acquisition is very costly and always involves some form of human involvement. By building a similar model with fewer input features, we can decrease the number of times data needs to be captured throughout the year. The cost saved associated with missing features can improve the value of our prediction model, as it is used by our JKI project partners. Moreover, feature selection is already part of most machine learning pipelines, also those unrelated to yield prediction. Our contribution to the discussion on Shapley values for feature selection can lead to a wider application of Shapley values for feature selection in random forests and maybe even replace the often used practice of considering the internal Gini feature importances for this problem, as they show a clearly worse performance in our experiments.

d) Gaps in remote sensing data

When trying to use remote sensing data for yield prediction with a high spatio-temporal resolution, cloud and cloud shadows occlude parts of the data necessary to train a model and make predictions. As the climatic conditions show some form of continuity in confined regions, we were able to present a novel deep interpolation approach to fix gaps in remote sensing data. We use partial convolutions within a U-Net deep learning architecture to interpolate remote sensing LST measurements from the MODIS sensors attached to NASA satellites. Our two-step interpolation approach includes first the conversion of air temperature data to LST data, when local ground-site weather stations can provide this information. Second, we use deep interpolation methods to achieve a gap-free representation of the data. Partial convolutions work like regular convolutions, but only focus on valid pixels and fill the remaining information when progressing deeper into the network. Furthermore, we deployed a partial loss function during training that again only evaluated valid pixels of the ground-truth data and allows training with exclusive use of occluded data. This is very important because the bigger the research area, the smaller the chances of finding completely non-occluded ground-truth data.

To the best of our knowledge, we are the first to explore the use of deep learning to fill LST remote sensing data. Regarding general image inpainting problems, our work uses partial convolutions (Liu et al. 2018) to fill irregular holes and the trusted idea of a U-Net design for the deep learning model (Ronneberger et al. 2015) and is therefore in-line with the current developments in the field. The problem of LST interpolation was previously handled using statistical methods (Metz et al. 2017), which have two major limitations compared to our deep interpolation approach. First, they are not able to fit an interpolation function when the number of accessible data points is too small. Second, they are not able to learn spatial dependencies of the interest region, as each instance is seen as an isolated problem.

Having established deep learning for inpainting of LST remote sensing data, we could consider other remote sensing products that can be used for yield prediction like infrared reflection to be filled by our deep interpolation approach. Having no access to consistently available pixels, like we have with weather station data for LST, poses an

additional challenge. A solution could be given by including a temporal axis to the process to provide a gapless coverage.

Finally, experiments in our research area show that we outperform the state-of-the-art statistical approaches. This is achieved through the capabilities of our approach in collecting knowledge of the research area's environmental constraints during training, which is missing for statistical approaches. This enables remote sensing LST measurements for plot-level yield prediction and is an important part of our grapevine yield prediction pipeline. As mentioned above, our work is not only useful for yield prediction scenarios, but can be of interest anywhere, where gapless remote sensing data can be used for enhanced predictions.

e) Making predictions for shifting domains

To conclude the challenges identified for yield prediction, we tested deep transfer learning as a tool to deal with shifting domains in yield prediction scenarios. We give a proof-of-concept for transfer learning that can lead to improved crop prediction using CNNs, in particular through a joint application of several regularization methods, together with careful determination of the hyperparameters. First, Batch Spectral Shrinking (BSS) and L^2-SP regularization provide more stable model training and improved prediction accuracy. In addition to the usual procedures of initializing the weights by a model trained on a larger data set and fixing these weights, transfer-specific regularization methods with simultaneous application of the Gaussian process lead to an improved prediction for soybean yield prediction in Argentina.

Transfer learning in yield prediction has only been done for very similar domains such as Argentina and Brazil (Wang et al. 2020) or by using a different crop in the same region (Khaki et al. 2021). Our contributions now allows for transfer learning for yield prediction on a world-wide scale to enable improved predictions. This is achieved by using state-of-the-art building blocks for transfer learning. The combination of BSS and L^2-SP outperformed any use of the also popular Delta regularization methods in our initial experiments.

To further build on our results, we could multiply the amount of data available for yield prediction if a model could be universally used for different crops. There might be universal patterns in remote sensing data that can be used to extract very low level features that might be applicable to a variety of crops, allowing to tackle the problem of small data sets from this direction. In addition, first efforts are made in our workgroup to remove the fixed 8-day intervals used in data preprocessing for remote sensing and instead use the idea of thermal time, where data are clustered according to the so-called growing degree days (DGG). The idea is to be able to better align the target and source domains by normalizing the time steps based on the phenological state of the target crop.

For grapevine yield prediction, the transfer of knowledge with deep learning is just a concept for now, as the amount of data is limited and it is difficult to train expressive deep learning models. A solution here is to analyze the trade-off between building multiple prediction models on homogenous variety selection with fewer training data each, or building a combined model for multiple varieties with more training data but also more variation within. The fast training times of XGBoost-based approaches allow

for a yearly adoption of the model for new data by being trained by the domain experts themselves. We developed the application for grapevine yield prediction with this idea in mind, and together with the tools for explainability allowing the domain experts to understand the model, they are able to provide reusability of the model, even in times of changing climatic conditions, by incorporating the new data every year. In general, the contributions regarding this challenge are rather yield prediction-specific but could be used as a starting point to explore other transfer learning problems based on remote sensing data.

Summarized this thesis contributions evolved around the challenges for explainable yield prediction, identified within the KI-iREPro project. We analyzed XGBoost as the go-to method for general yield prediction tasks, as even on the biggest data sets it is able to keep up with state-of-the-art deep learning approaches. At the same time, it provides faster training times together with better explainability. As the challenges of explainable yield prediction can be extended to machine learning in general, like explaining complex models with many features and using Shapley values feature importances for feature selection, so can the contributions of this thesis. Especially the extension of the well-known Shapley value idea for explaining models towards groups of features, as well as the CFS algorithm for feature selection, can be used for general machine learning tasks. The deep interpolation method for gaps in remote sensing LST data enables plot-level yield prediction, but can also be utilized in other tasks involving remote sensing data, as gapless coverage is needed in many research areas. Lastly, handling shifting domains appears in any kind of machine learning scenario. We were able to prove that a general transfer of knowledge is possible for worldwide yield prediction scenarios, even with our limited access to data. We are proud to find our methods applied in a real-world scenario within the KI-iREPro project, where viticulture experts use our solutions for modeling and explainability to improve their workflow.

Bibliography

Aas, K., Jullum, M. & Løland, A. (2020), 'Explaining individual predictions when features are dependent: More accurate approximations to shapley values', *preprint arXiv:1903.10464* .

Agrarmeteorologie RLP (2023), 'Ground-site weather station records in RLP', `https://www.wetter.rlp.de`. Data set - Accessed: 2023-10-27.

Akiba, T., Sano, S., Yanase, T., Ohta, T. & Koyama, M. (2019), Optuna: A next-generation hyperparameter optimization framework, *in* 'Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining', pp. 2623–2631.

Alem, A. & Kumar, S. (2022), 'Transfer learning models for land cover and land use classification in remote sensing image', *Applied Artificial Intelligence* **36**(1).

Alibabaei, K., Gaspar, P. D. & Lima, T. M. (2021), 'Crop yield estimation using deep learning based on climate big data and irrigation scheduling', *Energies* **14**(11).

Alsahaf, A., Petkov, N., Shenoy, V. & Azzopardi, G. (2022), 'A framework for feature selection through boosting', *Expert Systems with Applications* **187**.

Amoukou, S. I., Brunel, N. J. & Salaün, T. (2021), 'The shapley value of coalition of variables provides better explanations', *preprint arXiv:2103.13342* .

Arenas, M., Barceló, P., Romero Orth, M. & Subercaseaux, B. (2022), 'On computing probabilistic explanations for decision trees', *Advances in Neural Information Processing Systems* **35**, 28695–28707.

Aumann, R. & Shapley, L. (1974), 'Values of non atomic games', *Princeton University Press* .

Barnes, C., Shechtman, E., Finkelstein, A. & Goldman, D. B. (2009), 'Patchmatch: a randomized correspondence algorithm for structural image editing', *ACM SIGGRAPH 2009 papers* .

Bergstra, J., Yamins, D. & Cox, D. (2013), Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, *in* 'International conference on machine learning', PMLR, pp. 115–123.

Bóbeda, G. R., Fernández-Combarro Álvarez, E., Mazza, S., Giménez, L. I., Díaz Rodríguez, S. I. et al. (2018), 'Using regression trees to predict citrus load balancing accuracy and costs', *International Journal of Computational Intelligence, 12 (1)* .

Breiman, L. (2001), 'Random forests', *Machine learning* **45**(1), 5–32.

Bureau, U. C. (2018), 'Tiger: US Census counties 2018', `https://developers.google.com/earth-engine/datasets/catalog/TIGER_2018_Counties/`. Data set - Accessed: 2021-07-01.

Cao, J., Zhang, Z., Tao, F., Zhang, L., Luo, Y., Han, J. & Li, Z. (2020), 'Identifying the contributions of multi-source data for winter wheat yield prediction in China', *Remote Sensing* **12**(5).

Cao, J., Zhou, W., Zheng, Z., Ren, T. & Wang, W. (2021), 'Within-city spatial and temporal heterogeneity of air temperature and its relationship with land surface temperature', *Landscape and Urban Planning* **206**.

Castro, J., Gómez, D., Molina, E. & Tejada, J. (2017), 'Improving polynomial estimation of the shapley value by stratified random sampling with optimum allocation', *Computers & Operations Research* **82**, 180–188.

Castro, J., Gómez, D. & Tejada, J. (2009), 'Polynomial calculation of the shapley value based on sampling', *Computers & Operations Research* **36**(5), 1726–1730.

Celik, M. F., Isik, M. S., Taskin, G., Erten, E. & Camps-Valls, G. (2023), 'Explainable artificial intelligence for cotton yield prediction with multisource data', *IEEE Geoscience and Remote Sensing Letters* .

Chambers, R. G. & Pieralli, S. (2020), 'The sources of measured us agricultural productivity growth: Weather, technological change, and adaptation', *American Journal of Agricultural Economics* **102**(4), 1198–1226.

Chandrashekar, G. & Sahin, F. (2014), 'A survey on feature selection methods', *Computers & electrical engineering* **40**(1), 16–28.

Charoen-Ung, P. & Mittrapiyanuruk, P. (2018), Sugarcane yield grade prediction using random forest with forward feature selection and hyper-parameter tuning, *in* 'International Conference on Computing and Information Technology', Springer, pp. 33–42.

Chen, J., Sun, J., Li, Y. & Hou, C. (2022), 'Object detection in remote sensing images based on deep transfer learning', *Multimedia Tools and Applications* pp. 1–17.

Chen, T. & Guestrin, C. (2016), Xgboost: A scalable tree boosting system, *in* 'Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining', pp. 785–794.

Chen, X., Wang, S., Fu, B., Long, M. & Wang, J. (2019), 'Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning', *Advances in Neural Information Processing Systems* **32**.

Chu, B., Madhavan, V., Beijbom, O., Hoffman, J. & Darrell, T. (2016), Best practices for fine-tuning visual classifiers to new domains, *in* 'Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14', Springer, pp. 435–442.

Chu, C. C. F. & Chan, D. P. K. (2020), 'Feature selection using approximated high-order interaction components of the shapley value for boosted tree classifier', *IEEE Access* **8**, 112742–112750.

Cohen, S., Dror, G. & Ruppin, E. (2007), 'Feature selection via coalitional game theory', *Neural computation* **19**(7), 1939–1961.

Cohen, S., Ruppin, E. & Dror, G. (2005), 'Feature selection based on the shapley value', *other words* **1**(98Eqr).

Conitzer, V. & Sandholm, T. (2004), Computing shapley values, manipulating value division schemes, and checking core membership in multi-issue domains, *in* 'AAAI', Vol. 4, pp. 219–225.

Covert, I., Lundberg, S. M. & Lee, S.-I. (2020), 'Understanding global feature contributions with additive importance measures', *Advances in Neural Information Processing Systems* **33**, 17212–17223.

Craven, P. & Wahba, G. (1978), 'Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation', *Numerische mathematik* **31**(4), 377–403.

Dastour, H. & Hassan, Q. K. (2023), 'A comparison of deep transfer learning methods for land use and land cover classification', *Sustainability* **15**(10).

De La Fuente, M., Linares Torres, R., Baeza Trujillo, P., Miranda, C., Lissarrague Garcia-Gutierrez, J. R. et al. (2015), 'Comparison of different methods of grapevine yield prediction in the time window between fruitset and veraison', *Journal International des Sciences de la Vigne et du Vin* **49**(1), 27–35.

Deng, H. & Runger, G. (2012), Feature selection via regularized trees, *in* 'The 2012 International Joint Conference on Neural Networks (IJCNN)', IEEE, pp. 1–8.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009), Imagenet: A large-scale hierarchical image database, *in* '2009 IEEE conference on computer vision and pattern recognition', Ieee, pp. 248–255.

Desloires, J., Ienco, D. & Botrel, A. (2023), 'Out-of-year corn yield prediction at field-scale using sentinel-2 satellite imagery and machine learning methods', *Computers and Electronics in Agriculture* **209**.

Dhal, P. & Azad, C. (2022), 'A comprehensive survey on feature selection in the various fields of machine learning', *Applied Intelligence* **52**(4), 4543–4581.

Díaz, I., Mazza, S., Combarro, E. & Gimenez, L. (2017), 'Machine learning applied to the prediction of citrus production', *Spanish journal of agricultural research* **15**.

Fanaee-T, H. (2013), 'Bike Sharing Dataset', `https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset`. Data set - Accessed: 2023-08-23.

Fang, L., Jin, J., Segers, A., Lin, H. X., Pang, M., Xiao, C., Deng, T. & Liao, H. (2022), 'Development of a regional feature selection-based machine learning system for air pollution forecasting over china', *Geoscientific Model Development* **15**(20), 7791–7807.

Fernandez-Beltran, R., Baidar, T., Kang, J. & Pla, F. (2021), 'Rice-yield prediction with multi-temporal sentinel-2 data and 3D CNN: A case study in Nepal', *Remote Sensing* **13**(7).

Flores, R., Molina, E. & Tejada, J. (2019), 'Evaluating groups with the generalized shapley value', *4OR, A Quarterly Journal of Operations Research* **17**(2), 141–172.

Fournier-Viger, P. (2016), 'The data mining blog: The KDDCup 2015 dataset', `https://data-mining.philippe-fournier-viger.com/the-kddcup-2015-dataset-download-link/`. Accessed: 2022-02-01.

Friedl, M., Sulla-Menashe, D. (2019), 'MCD12Q1 MODIS/terra+aqua land cover type yearly l3 global 500m sin grid v006', `https://doi.org/10.5067/MODIS/MCD12Q1.006`. Data set - Accessed: 2022-11-11.

Friedman, J. H. (2001), 'Greedy function approximation: a gradient boosting machine', *Annals of statistics* pp. 1189–1232.

Fryer, D., Strümke, I. & Nguyen, H. (2021), 'Shapley values for feature selection: The good, the bad, and the axioms', *Ieee Access* **9**, 144352–144360.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., March, M. & Lempitsky, V. (2016), 'Domain-adversarial training of neural networks', *Journal of machine learning research* **17**(59), 1–35.

Gao, B.-C. (1996), 'NDWI—a normalized difference water index for remote sensing of vegetation liquid water from space', *Remote sensing of environment* **58**(3), 257–266.

Garcia, D. (2010), 'Robust smoothing of gridded data in one and higher dimensions with missing values', *Computational Statistics and Data Analysis* **54**(4), 1167–1178.

Garreau, D. & Luxburg, U. (2020), Explaining the explainer: A first theoretical analysis of LIME, *in* 'International conference on artificial intelligence and statistics', PMLR, pp. 1287–1296.

Gazzola, G. & Jeong, M. K. (2019), 'Dependence-biased clustering for variable selection with random forests', *Pattern Recognition* **96**.

Genuer, R., Poggi, J.-M. & Tuleau-Malot, C. (2010), 'Variable selection using random forests', *Pattern recognition letters* **31**(14), 2225–2236.

Gohain, K. J., Mohammad, P. & Goswami, A. (2021), 'Assessing the impact of land use land cover changes on land surface temperature over Pune city, India', *Quaternary International* **575**, 259–269.

Goldstein, A., Kapelner, A., Bleich, J. & Pitkin, E. (2015), 'Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation', *journal of Computational and Graphical Statistics* **24**(1), 44–65.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D. & Moore, R. (2017), 'Google earth engine: Planetary-scale geospatial analysis for everyone', *Remote sensing of Environment* **202**, 18–27.

Grabisch, M. & Roubens, M. (1999), 'An axiomatic approach to the concept of interaction among players in cooperative games', *International Journal of game theory* **28**(4), 547–565.

Gupta, J., Pathak, S. & Kumar, G. (2022), Deep learning (CNN) and transfer learning: A review, *in* 'Journal of Physics: Conference Series', Vol. 2273, IOP Publishing.

Han, B. & Howe, B. (2023), 'Adapting to skew: Imputing spatiotemporal urban data with 3d partial convolutions and biased masking', *preprint arXiv:2301.04233* .

Harrison, D. & Rubinfeld, D. (1978), 'Hedonic prices and the demand for clean air', `https://www.kaggle.com/datasets/schirmerchad/bostonhoustingmlnd"`. Data Set - Accessed 2023-11-20.

He, K., Zhang, X., Ren, S. & Sun, J. (2016), Deep residual learning for image recognition, *in* 'Proceedings of the IEEE conference on computer vision and pattern recognition', pp. 770–778.

Hilal, A., Al-Wesabi, F., Alzahrani, K., Al Duhayyim, M., Hamza, M., Rizwanullah, M. & Díaz, V. (2022), 'Deep transfer learning based fusion model for environmental remote sensing image classification model', *European Journal of Remote Sensing* **55**, 1–12.

Huan, L. & Hiroshi, M. (1998), *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic Publishers, New York.

Huang, N., Lu, G. & Xu, D. (2016), 'A permutation importance-based feature selection method for short-term electricity load forecasting using random forest', *Energies* **9**(10).

Huang, X. & Marques-Silva, J. (2023), 'The inadequacy of shapley values for explainability', *preprint arXiv:2302.08160* .

Huber, F., Engler, H., Kicherer, A., Herzog, K., Töpfer, R. & Steinhage, V. (2023), Grouping shapley value feature importances of random forests for explainable yield prediction, *in* 'Intelligent Systems and Applications', Springer Nature Switzerland, pp. 210–228.

Huber, F., Hofmann, B., Engler, H., Gauweiler, P., Fischer, B., Herzog, K., Kicherer, A., Töpfer, R., Gruna, R. & Steinhage, V. (2024c), 'A concept study for feature extraction and modeling for grapevine yield prediction', *Vitis* **63**.

Huber, F., Inderka, A. & Steinhage, V. (2024a), 'Leveraging remote sensing data for yield prediction with deep transfer learning', *Sensors* **24**(3).

Huber, F., Schulz, S. & Steinhage, V. (2024b), 'Deep interpolation of remote sensing land surface temperature data with partial convolutions', *Sensors* **24**(5).

Huber, F. & Steinhage, V. (2024d), 'Conditional feature selection: Evaluating model averaging when selecting features with shapley values', *Geomatics* **4**(3), 286–310.

Huber, F., Yushchenko, A., Stratmann, B. & Steinhage, V. (2022), 'Extreme gradient boosting for yield estimation compared with deep learning approaches', *Computers and Electronics in Agriculture* **202**.

Huh, M., Agrawal, P. & Efros, A. A. (2016), 'What makes imagenet good for transfer learning?', *preprint arXiv:1608.08614* .

Iman, M., Rasheed, K. & Arabnia, H. R. (2022), 'A review of deep transfer learning and recent advancements', *preprint arXiv:2201.09679* .

Inderka, A., Huber, F. & Steinhage, V. (2024), 'On convolutional vision transformers for yield prediction', *preprint arXiv:2402.05557* .

Jacovi, A., Swayamdipta, S., Ravfogel, S., Elazar, Y., Choi, Y. & Goldberg, Y. (2021), 'Contrastive explanations for model interpretability', *preprint arXiv:2103.01378* .

Jiang, S., Mao, H., Ding, Z. & Fu, Y. (2019), 'Deep decision tree transfer boosting', *IEEE transactions on neural networks and learning systems* **31**(2), 383–395.

Jijón-Palma, M. E., Amisse, C. & Centeno, J. A. S. (2023), 'Hyperspectral dimensionality reduction based on sae-1dcnn feature selection approach', *Applied Geomatics* **15**(4), 991–1004.

Jullum, M., Redelmeier, A. & Aas, K. (2021), 'groupshapley: efficient prediction explanation with shapley values for feature groups', *preprint arXiv:2106.12228* .

Kaneko, A., Kennedy, T., Mei, L., Sintek, C., Burke, M., Ermon, S. & Lobell, D. (2019), Deep learning for crop yield prediction in Africa, *in* 'ICML workshop on artificial intelligence for social good', pp. 33–37.

Khaki, S., Pham, H. & Wang, L. (2021), 'Simultaneous corn and soybean yield prediction from remote sensing data using deep transfer learning', *Scientific Reports* **11**(1), 1–14.

Khaki, S. & Wang, L. (2019), 'Crop yield prediction using deep neural networks', *Frontiers in plant science* **10**.

Kingma, D. P. & Ba, J. (2014), 'Adam: A method for stochastic optimization', *preprint arXiv:1412.6980* .

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A. et al. (2017), 'Overcoming catastrophic forgetting in neural networks', *Proceedings of the national academy of sciences* **114**(13), 3521–3526.

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S. & Houlsby, N. (2020), Big transfer (bit): General visual representation learning, *in* 'European conference on computer vision', Springer, pp. 491–507.

Krug, D. (2018), ': Soil map of germany 1:250,000 (buek250)'. Data set - Bundesanstalt für Geowissenschaften und Rohstoffe (BGR), Hannover.

Kumar, I. E., Venkatasubramanian, S., Scheidegger, C. & Friedler, S. (2020), Problems with shapley-value-based explanations as feature importance measures, *in* 'International Conference on Machine Learning', PMLR, pp. 5491–5500.

Lam, S. K., Pitrou, A. & Seibert, S. (2015), Numba: A llvm-based python jit compiler, *in* 'Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC', pp. 1–6.

Li, B., Liang, S., Liu, X., Ma, H., Chen, Y., Liang, T. & He, T. (2021), 'Estimation of all-sky 1 km land surface temperature over the conterminous united states', *Remote Sensing of Environment* **266**.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J. & Liu, H. (2017), 'Feature selection: A data perspective', *ACM computing surveys (CSUR)* **50**(6), 1–45.

Li, W., Wang, Z., Wang, Y., Wu, J., Wang, J., Jia, Y. & Gui, G. (2020), 'Classification of high-spatial-resolution remote sensing scenes method using transfer learning and deep convolutional neural network', *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **13**, 1986–1995.

Li, X., Xiong, H., Wang, H., Rao, Y., Liu, L., Chen, Z. & Huan, J. (2019), 'Delta: Deep learning transfer using feature map with attention for convolutional networks', *preprint arXiv:1901.09229* .

Li, X., Zhou, Y., Asrar, G. R. & Zhu, Z. (2018), 'Creating a seamless 1 km resolution daily land surface temperature dataset for urban and surrounding areas in the conterminous United States', *Remote Sensing of Environment* **206**, 84–97.

Linardatos, P., Papastefanopoulos, V. & Kotsiantis, S. (2020), 'Explainable ai: A review of machine learning interpretability methods', *Entropy* **23**(1).

Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A. & Catanzaro, B. (2018), Image inpainting for irregular holes using partial convolutions, *in* 'The European Conference on Computer Vision (ECCV)'.

Liu, H., Lu, N., Jiang, H., Qin, J. & Yao, L. (2020), 'Filling gaps of monthly terra/modis daytime land surface temperature using discrete cosine transform method', *Remote Sensing* **12**(3).

Lobell, D. B. & Burke, M. B. (2010), 'On the use of statistical models to predict crop yield responses to climate change', *Agricultural and forest meteorology* **150**(11), 1443–1452.

Louppe, G., Wehenkel, L., Sutera, A. & Geurts, P. (2013), 'Understanding variable importances in forests of randomized trees', *Advances in neural information processing systems* **26**.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N. & Lee, S.-I. (2020), 'From local explanations to global understanding with explainable AI for trees', *Nature machine intelligence* **2**(1), 56–67.

Lundberg, S. M. & Lee, S.-I. (2017), A unified approach to interpreting model predictions, *in* 'Advances in Neural Information Processing Systems 30', Curran Associates, Inc., pp. 4765–4774.

Ma, Y., Chen, S., Ermon, S. & Lobell, D. B. (2024), 'Transfer learning in environmental remote sensing', *Remote Sensing of Environment* **301**.

Ma, Y., Yang, Z., Huang, Q. & Zhang, Z. (2023), 'Improving the transferability of deep learning models for crop yield prediction: A partial domain adaptation approach', *Remote Sensing* **15**(18).

Madigan, D. & Raftery, A. E. (1994), 'Model selection and accounting for model uncertainty in graphical models using occam's window', *Journal of the American Statistical Association* **89**(428), 1535–1546.

Man, X. & Chan, E. (2021), 'The best way to select features? comparing mda, lime, and shap', *The Journal of Financial Data Science Winter* **3**(1), 127–139.

Marcílio, W. E. & Eler, D. M. (2020), From explanations to feature selection: assessing shap values as feature selection mechanism, *in* '2020 33rd SIBGRAPI conference on Graphics, Patterns and Images (SIBGRAPI)', Ieee, pp. 340–347.

Marichal, J.-L., Kojadinovic, I. & Fujimoto, K. (2007), 'Axiomatic characterizations of generalized values', *Discrete Applied Mathematics* **155**(1), 26–43.

Martínez-Ferrer, L., Piles, M. & Camps-Valls, G. (2020), 'Crop yield estimation and interpretability with Gaussian processes', *IEEE Geoscience and Remote Sensing Letters* **18**(12), 2043–2047.

Meng, L., Liu, H., L Ustin, S. & Zhang, X. (2021), 'Predicting maize yield at the plot scale of different fertilizer systems by multi-source data and machine learning methods', *Remote Sensing* **13**(18).

Meng, L., Liu, H., Zhang, X., Ren, C., Ustin, S., Qiu, Z., Xu, M. & Guo, D. (2019), 'Assessment of the effectiveness of spatiotemporal fusion of multi-source satellite images for cotton yield estimation', *Computers and Electronics in Agriculture* **162**, 44–52.

Menze, B. H., Kelm, B. M., Masuch, R., Himmelreich, U., Bachert, P., Petrich, W. & Hamprecht, F. A. (2009), 'A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data', *BMC bioinformatics* **10**, 1–16.

Metz, M., Andreo, V. & Neteler, M. (2017), 'A new fully gap-free time series of land surface temperature from modis lst data', *Remote Sensing* **9**(12).

Ministerio de Agricultura (2023), 'Soja - serie siembra, cosecha, producción, rendimiento', `https://datosestimaciones.magyp.gob.ar/reportes.php?reporte=Estimaciones`. Data set - Accessed: 2023-05-09.

Mollas, I., Bassiliades, N., Vlahavas, I. & Tsoumakas, G. (2019), 'Lionforests: local interpretation of random forests', *preprint arXiv:1911.08780* .

Mylonas, N., Mollas, I., Bassiliades, N. & Tsoumakas, G. (2022), Local multi-label explanations for random forest, *in* 'Joint European Conference on Machine Learning and Knowledge Discovery in Databases', Springer, pp. 369–384.

Nathgosavi, V. & Patil, S. (2021), 'A survey on crop yield prediction using machine learning', *Turkish Journal of Computer and Mathematics Education* **12**(13), 2343–2347.

Nembrini, S., König, I. R. & Wright, M. N. (2018), 'The revival of the gini importance?', *Bioinformatics* **34**(21), 3711–3718.

Neteler, M. (2010), 'Estimating daily land surface temperatures in mountainous environments by reconstructed modis lst data', *Remote sensing* **2**(1), 333–351.

Nori, H., Jenkins, S., Koch, P. & Caruana, R. (2019), 'Interpretml: A unified framework for machine learning interpretability', *preprint arXiv:1909.09223* .

Okuta, R., Unno, Y., Nishino, D., Hido, S. & Loomis, C. (2017), Cupy: A numpy-compatible library for nvidia gpu calculations, *in* 'Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)'.

Palliotti, A. & Cartechini, A. (1998), Cluster thinning effects on yield and grape composition in different grapevine cultivars, *in* 'XXV International Horticultural Congress, Part 2: Mineral Nutrition and Grape and Wine Quality 512', pp. 111–120.

Pan, S. J. & Yang, Q. (2009), 'A survey on transfer learning', *IEEE Transactions on knowledge and data engineering* **22**(10), 1345–1359.

Park, M. S., Na, J. H. & Choi, J. Y. (2005), Pca-based feature extraction using class information, *in* '2005 IEEE International Conference on Systems, Man and Cybernetics', Vol. 1, IEEE, pp. 341–345.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. & Efros, A. A. (2016), 'Context encoders: Feature learning by inpainting', *preprint arXiv:1604.07379* .

Pham, H., Dai, Z., Xie, Q. & Le, Q. V. (2021), Meta pseudo labels, *in* 'Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition', pp. 11557–11568.

Pham, H. T., Kim, S., Marshall, L. & Johnson, F. (2019), 'Using 3d robust smoothing to fill land surface temperature gaps at the continental scale', *International Journal of Applied Earth Observation and Geoinformation* **82**.

Plested, J. & Gedeon, T. (2022), 'Deep transfer learning for image classification: a survey', *preprint arXiv:2205.09904* .

Prosekov, A. Y. & Ivanova, S. A. (2018), 'Food security: The challenge of the present', *Geoforum* **91**, 73–77.

Quarmby, N., Milnes, M., Hindle, T. & Silleos, N. (1993), 'The use of multi-temporal NDVI measurements from AVHRR data for crop yield estimation and prediction', *International Journal of Remote Sensing* **14**(2), 199–210.

Raftery, A. E. (1995), 'Bayesian model selection in social research', *Sociological methodology* pp. 111–163.

Redelmeier, A., Jullum, M. & Aas, K. (2020), 'Explaining predictive models with mixed features using shapley values and conditional inference trees', *preprint arXiv:2007.01027* .

Ribeiro, M. T., Singh, S. & Guestrin, C. (2016), " why should i trust you?" explaining the predictions of any classifier, *in* 'Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining', pp. 1135–1144.

Ribeiro, M. T., Singh, S. & Guestrin, C. (2018), Anchors: High-precision model-agnostic explanations, *in* 'Proceedings of the AAAI conference on artificial intelligence', Vol. 32.

Rodríguez, S. I. D., Mazza, S. M., Álvarez, E. F. C., Giménez, L. I. & Gaiad, J. E. (2017), 'Machine learning applied to the prediction of citrus production', *Spanish journal of agricultural research* **15**(2).

Ronneberger, O., Fischer, P. & Brox, T. (2015), U-net: Convolutional networks for biomedical image segmentation, *in* 'Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015', Springer International Publishing, pp. 234–241.

Roth, A. E. (1988), *The Shapley value: essays in honor of Lloyd S. Shapley*, Cambridge University Press.

Rozemberczki, B., Watson, L., Bayer, P., Yang, H.-T., Kiss, O., Nilsson, S. & Sarkar, R. (2022), 'The shapley value in machine learning', *preprint arXiv:2202.05594* .

Sathishkumar, V. & Cho, Y. (2020), 'Season wise bike sharing demand analysis using random forest algorithm', *Computational Intelligence* .

Schneider, P. & Hook, S. J. (2010), 'Space observations of inland water bodies show rapid surface warming since 1985', *Geophysical Research Letters* **37**(22).

Segev, N., Harel, M., Mannor, S., Crammer, K. & El-Yaniv, R. (2016), 'Learn on source, refine on target: A model transfer learning framework with random forests', *IEEE transactions on pattern analysis and machine intelligence* **39**(9), 1811–1824.

Selbst, A. D. & Barocas, S. (2018), 'The intuitive appeal of explainable machines', *Fordham L. Rev.* **87**.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. & Batra, D. (2017), Grad-cam: Visual explanations from deep networks via gradient-based localization, *in* 'Proceedings of the IEEE international conference on computer vision', pp. 618–626.

Sharif Razavian, A., Azizpour, H., Sullivan, J. & Carlsson, S. (2014), CNN features off-the-shelf: an astounding baseline for recognition, *in* 'Proceedings of the IEEE conference on computer vision and pattern recognition workshops', pp. 806–813.

Shih, A., Choi, A. & Darwiche, A. (2018), 'A symbolic approach to explaining bayesian network classifiers', *preprint arXiv:1805.03364* .

Simonyan, K., Vedaldi, A. & Zisserman, A. (2013), 'Deep inside convolutional networks: Visualising image classification models and saliency map', *preprint arXiv:1312.6034* .

Sirsat, M. S., Mendes-Moreira, J., Ferreira, C. & Cunha, M. (2019), 'Machine learning predictive model of grapevine yield based on agroclimatic patterns', *Engineering in Agriculture, Environment and Food* **12**(4), 443–450.

Sobel, I. (2014), 'An isotropic 3x3 image gradient operator', *Presentation at Stanford A.I. Project 1968* .

Strumbelj, E. & Kononenko, I. (2010), 'An efficient explanation of individual classifications using game theory', *The Journal of Machine Learning Research* **11**, 1–18.

Štrumbelj, E. & Kononenko, I. (2014), 'Explaining prediction models and individual predictions with feature contributions', *Knowledge and information systems* **41**, 647–665.

Sun, H., Ma, J., Guo, Q., Zou, Q., Song, S., Lin, Y. & Yu, H. (2022), 'Coarse-to-fine task-driven inpainting for geoscience images', *preprint arXiv:2211.11059* .

Sun, J., Di, L., Sun, Z., Shen, Y. & Lai, Z. (2019), 'County-level soybean yield prediction using deep CNN-LSTM model', *Sensors* **19**(20).

Sundararajan, M. & Najmi, A. (2020), The many shapley values for model explanation, *in* 'International conference on machine learning', PMLR, pp. 9269–9278.

Sundararajan, M., Taly, A. & Yan, Q. (2017), Axiomatic attribution for deep networks, *in* 'International conference on machine learning', PMLR, pp. 3319–3328.

Thornton, P., Thornton, M., Mayer, B., Wei, Y., Devarakonda, R., Vose, R. & Cook, R. (2016), 'Daymet: daily surface weather data on a 1-km grid for North America, version 3. ORNL DAAC, Oak Ridge, Tennessee, USA'.

Töpfer, R. & Trapp, O. (2022), 'A cool climate perspective on grapevine breeding: climate change and sustainability are driving forces for changing varieties in a traditional market', *Theoretical and Applied Genetics* **135**(11), 3947–3960.

Tseng, G. (2022), 'pycrop-yield-prediction', `https://github.com/gabrieltseng/pycrop-yield-prediction`. Accessed: 2022-02-01.

Tseng, H.-H., Yang, M.-D., Saminathan, R., Hsu, Y.-C., Yang, C.-Y. & Wu, D.-H. (2022), 'Rice seedling detection in uav images using transfer learning and machine learning', *Remote Sensing* **14**(12).

USDA (2010), *Field Crops. Agricultural Handbook Number (628)*, USDA.

USDA (2021), 'USDA nass quick stats database', `https://quickstats.nass.usda.gov/`. Data set - Accessed: 2021-07-01.

USDA (2023), 'Country Summary', `https://ipad.fas.usda.gov/countrysummary/`. Data set - Accessed: 2023-2-21.

Van Klompenburg, T., Kassahun, A. & Catal, C. (2020), 'Crop yield prediction using machine learning: A systematic literature review', *Computers and Electronics in Agriculture* **177**.

Venkatesh, B. & Anuradha, J. (2019), 'A review of feature selection and its methods', *Cybernetics and information technologies* **19**(1), 3–26.

Vermote, E. (2021), 'MODIS/Terra Surface Reflectance 8-Day L3 Global 500m SIN Grid V061. *NASA EOSDIS Land Processes DAAC.*', `https://doi.org/10.5067/MODIS/MOD09A1.061`. Data set - Accessed: 2022-10-26.

Vito, S. (2016), 'Air Quality', `https://archive.ics.uci.edu/dataset/360/air+quality`. Data set - Accessed 23-08-23.

Wachter, S., Mittelstadt, B. & Russell, C. (2017), 'Counterfactual explanations without opening the black box: Automated decisions and the gdpr', *Harv. JL & Tech.* **31**.

Wan, Z. & Dozier, J. (1996), 'A generalized split-window algorithm for retrieving land-surface temperature from space', *IEEE Transactions on geoscience and remote sensing* **34**(4), 892–905.

Wan, Z., Hook, S. & Hulley, G. (2023), 'Modis/terra land surface temperature/emissivity daily l3 global 1km sin grid v061'. Data Set - Accessed: 2023-10-27.

Wang, A. X., Tran, C., Desai, N., Lobell, D. & Ermon, S. (2018), Deep transfer learning for crop yield prediction with remote sensing data, *in* 'Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies', pp. 1–5.

Wang, X., Huang, J., Feng, Q. & Yin, D. (2020), 'Winter wheat yield prediction at county level and uncertainty analysis in main wheat-producing regions of China with deep learning approaches', *Remote Sensing* **12**(11).

Wang, Y. & Witten, I. H. (1996), 'Induction of model trees for predicting continuous classes', *Computer Science Working Papers* .

Wolanin, A., Mateo-García, G., Camps-Valls, G., Gómez-Chova, L., Meroni, M., Duveiller, G., Liangzhi, Y. & Guanter, L. (2020), 'Estimating and understanding crop yields with explainable deep learning in the Indian wheat belt', *Environmental Research Letters* **15**(2).

Xiao, Y., Li, S., Huang, J., Huang, R. & Zhou, C. (2023), 'A new framework for the reconstruction of daily 1 km land surface temperatures from 2000 to 2022', *Remote Sensing* **15**(20).

Xu, L., Skoularidou, M., Cuesta-Infante, A. & Veeramachaneni, K. (2019), 'Modeling tabular data using conditional gan', *Advances in Neural Information Processing Systems* **32**.

Xu, Y. & Shen, Y. (2013), 'Reconstruction of the land surface temperature time series using harmonic analysis', *Computers & geosciences* **61**, 126–132.

Xuhong, L., Grandvalet, Y. & Davoine, F. (2018), Explicit inductive bias for transfer learning with convolutional networks, *in* 'International Conference on Machine Learning', PMLR, pp. 2825–2834.

Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O. & Li, H. (2017), 'High-resolution image inpainting using multi-scale neural patch synthesis', *preprint arXiv:1611.09969* .

Yang, L., Song, W., Xu, C., Sapey, E., Jiang, D. & Wu, C. (2023), 'Effects of high night temperature on soybean yield and compositions', *Frontiers in Plant Science* **14**.

You, J., Li, X., Low, M., Lobell, D. & Ermon, S. (2017), 'Deep Gaussian process for crop yield prediction based on remote sensing data', *Thirty-First AAAI conference on artificial intelligence* .

Zacharias, J., von Zahn, M., Chen, J. & Hinz, O. (2022), 'Designing a feature selection method based on explainable artificial intelligence', *Electronic Markets* **32**(4), 2159–2184.

Zafar, M. R. & Khan, N. M. (2019), 'Dlime: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems', *preprint arXiv:1906.10263* .

Zeiler, M. D., Krishnan, D., Taylor, G. W. & Fergus, R. (2010), Deconvolutional networks, *in* '2010 IEEE Computer Society Conference on computer vision and pattern recognition', IEEE, pp. 2528–2535.

Zhang, T., Zhou, Y., Zhu, Z., Li, X. & Asrar, G. R. (2022), 'A global seamless 1 km resolution daily land surface temperature dataset (2003–2020)', *Earth System Science Data* **14**(2), 651–664.

Zhao, Y., Potgieter, A. B., Zhang, M., Wu, B. & Hammer, G. L. (2020), 'Predicting wheat yield at the field scale by combining high-resolution sentinel-2 satellite imagery and crop modelling', *Remote Sensing* **12**(6).

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. & Torralba, A. (2016), Learning deep features for discriminative localization, *in* 'Proceedings of the IEEE conference on computer vision and pattern recognition', pp. 2921–2929.

Zhou, H., Zhang, J., Zhou, Y., Guo, X. & Ma, Y. (2021), 'A feature selection algorithm of decision tree based on feature weight', *Expert Systems with Applications* **164**.

Zhou, K., Liu, Z., Qiao, Y., Xiang, T. & Loy, C. C. (2022), 'Domain generalization: A survey', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45**(4), 4396–4415.